

DONNÉES TECHNIQUES

Outils et méthodes du canulier

Édition juin 2026

David Bories

Données techniques - Outils et méthodes pour un canalier

Conçu et dirigé par David Bories — Albi, France.
Rédigé en utilisant l'intelligence artificielle générative.

Octobre 2026.

0.12.0-poche

contact@artisanat-de-la-donnee.fr

Table des matières

Avant-propos — Un manuel de forme	1
Ce que ce livre est, et ce qu'il n'est pas	1
Pour qui	2
Comment ce manuel est organisé	2
La règle de ce manuel en une phrase	2
Chapitre 1 — Le canal, le canulier, l'œuvre	3
Le fleuve, le canal, la donnée	3
L'intention et les critères	3
L'optimum observationnel	4
Les trois temps, les trois gestes	4
L'œuvre du canulier	5
La posture de référent	5
Chapitre 2 — La méthode hexagonale : creuser du pérenne	7
Le principe : isoler le cœur	7
Pourquoi c'est l'outil de la pérennité	8
L'infrastructure obéit aux ports	8
Le geste du canulier : modéliser avant de coder	9
L'organisation concrète d'un projet hexagonal	11
Ce que l'hexagone n'isole pas tout seul	11
Chapitre 3 — Le cœur sans dépendance : HTML, CSS, JavaScript	13
Pourquoi le natif plutôt qu'un framework	13
Les trois matériaux, et ce que chacun porte	14
Application web, application installable : le même cœur	14
Le stockage local du cœur	15
Plusieurs établis, une seule œuvre	15
Connecter le monde extérieur — toujours par l'hexagone	15
Une note d'honnêteté	16
Chapitre 4 — La gamification : rendre l'œuvre agréable et engageante	17
Le principe : transformer la donnée consolidée en signal motivant	17
Les mécaniques, et le geste métier qu'elles servent	18

Les seuils : le pont entre gamification et pilotage	18
Le garde-fou : gamifier sans falsifier	19
L'expérience fluide, au-delà des mécaniques	19
Chapitre 5 — Fabriquer la donnée : prélèvement et annotation	21
Cartographier la matière : quelles données fabriquer	21
Régler la durée limite d'utilisation	22
L'armature et le prélèvement : le lit et l'eau	22
La liste d'adjacence : tenir l'arbre	23
Les deux arêtes : la portée et la référence	23
Imbriquer ou lister : le critère est la recherche	24
L'armature s'entretient	24
Les outils de prélèvement	25
Les outils d'annotation	27
Régler le prélèvement à l'optimum	27
Documenter ses instruments	28
Chapitre 6 — Faire circuler : fixation et transport	29
Les outils de fixation : où la donnée tient	29
Le système d'événements : faire circuler l'acte d'inscription . . .	30
Le transport local : piloter son activité	31
Le transport distant : partager la donnée	32
La multiplication : un fait partagé n'est plus un objet unique . .	33
Réutiliser sans transmettre : le moment objectal, et comment le fond le tient	33
Le flux entrant : prélever le courrier	34
Chapitre 7 — Lire la donnée : le tableau de bord de pilotage	37
Le lecteur est situé, sa fenêtre est finie	37
La sélectivité : moins, mais juste	37
Les indicateurs : choisir les bons	38
La fraîcheur de la lecture : irriguer le tableau de bord	38
Les outils de lecture, et leurs biais	39
Les trois postures du lecteur, et ce qu'elles dictent à l'interface .	40
Entretenir la capacité de lire	40
La fenêtre nommée, dans la signature	40
Chapitre 8 — L'IA locale : l'opérateur sémantique du canulier	41
Ce que l'IA générative sait faire pour le solopreneur	41
Pourquoi local : souveraineté, coût, maîtrise	42
Combinaison l'IA et les règles déterministes	42
La méthode d'automatisation progressive	43
La contrepartie, nommée	43
Chapitre 9 — Entretenir : maintenance, sécurité, rituels	45
Les outils de maintenance	45
La sécurité et l'hygiène numérique	46
La réserve : le double retiré du cycle d'usage	46
Les rituels de l'atelier	47

La signature de l'œuvre	48
Documentation vivante : la maintenance par le client	48
Chapitre 10 — L'atelier numérique : le matériel et les logiciels	51
Le matériel	51
Les logiciels	52
L'atelier comme lieu, physique ou virtuel	53
L'articulation atelier / chaîne	54
Chapitre 11 — Parcours d'équipement : assembler les outils	55
Le procédé, toujours le même : trois gestes	55
Le parcours type, étape par étape	55
Trois assemblages illustrés	57
Le fil qui tient tout	57
Chapitre 12 — L'atelier d'agents IA : industrialiser le parcours	59
Le socle technique : ne pas refaire l'éprouvé	59
Les Agents IA	60
Deux objets à ne pas confondre : l'atelier d'agents IA et l'œuvre	60
L'agent de codage : l'établi des agents	60
Le dépôt : sauvegarde, mémoire, et base distribuée	61
Les ports souverains : étendre sans dépendre	61
Le pilotage de l'atelier : déterministe d'abord, l'humain aux jalons	61
La mise en pratique	62
Ce qui, dans ce chapitre, ne se périme pas	62
Annexe — La boîte à outils du canalier	63
A. Les six familles d'outils, en un coup d'œil	63
B. Les méthodes transversales	63
C. Checklist du flux d'observation rencontré	64
D. Grille de péremption	64
E. Modèle de revue de canal (ordre du jour)	64
F. Fiche de signature d'œuvre	65
G. Aide-mémoire matériel et logiciel	65
H. Les garde-fous, en sept phrases	65
I. Table de correspondance — le vocabulaire du corpus et les cadres existants (forme datée, juillet 2026)	66

Avant-propos — Un manuel de forme

« Le concept est une lampe ; il ne dessine pas la carte des récifs. » Toute la trilogie *À l'épreuve du flux* s'est tenue à cette règle : dire ce qui est vrai de la donnée pour longtemps, et laisser à d'autres le soin de déterminer ce qu'il faut en faire aujourd'hui. Le présent ouvrage fait le pari inverse, et l'assume. Il est un livre de **forme**.

Là où *Le fleuve et le canal* posait qu'il faut creuser des canaux dans le flux du réel, là où *Artisan de la donnée* nommait le canalier et ses six familles d'outils sans en prescrire l'usage concret, ce manuel descend d'un cran : il dit **avec quoi** et **comment** on creuse, aujourd'hui, en 2026, un canal numérique. Quels langages, quelles machines, quelles architectures, quels gestes. Il est l'établi du canalier, ouvert et garni.

Ce que ce livre est, et ce qu'il n'est pas

Ce livre **est** un manuel d'utilisation. Il décrit des outils matériels et logiciels disponibles maintenant, des méthodes de fabrication éprouvées, et la manière de les tenir au service d'un seul but : offrir au client — le solopreneur, l'entrepreneur individuel — une gestion d'activité fluide, lisible, maîtrisée, et la sérénité qui vient avec.

Ce livre **n'est pas** une doctrine. Tout ce qu'il nomme est daté. Les langages cités vivront et mourront ; les modèles d'IA locaux d'aujourd'hui seront remplacés ; les versions changeront. C'est sans importance, à une condition : que le lecteur sache distinguer en permanence **le fond** (ce qui ne se périmé pas — la posture de référent, la justesse du prélèvement, la responsabilité de l'œuvre) de **la forme** (les outils, les langages, les plateformes). Le fond est hérité de la trilogie ; ce manuel ne le rejoue pas, il s'en sert. La forme est ce qu'il ajoute, en sachant qu'elle est révisable.

Quand ce manuel écrit « JavaScript », « Docker » ou « Ollama », il faut lire ces mots comme on lirait, dans un texte de 1950, « une automobile peut atteindre 100 km/h » : vrai à la date d'écriture, accessoire au raisonnement, destiné à être dépassé. Ce qui engage le manuel, c'est

le *principe* que l'outil sert — la séparation du cœur métier et de l'infrastructure, la maîtrise locale, la pérennité du support — pas la marque de l'outil.

Pour qui

Pour le **canalier** — l'artisan de la donnée — qui veut équiper son atelier numérique. Pour l'apprenti qui apprend les gestes. Pour le solopreneur curieux qui veut comprendre ce que son outil fait, et pourquoi. Le manuel suppose acquise la trilogie : il n'y revient qu'en rappels resserrés, là où c'est nécessaire pour tenir un geste au clair.

Comment ce manuel est organisé

L'ouvrage suit l'ossature donnée par le *Cahier de l'atelier* : les **six familles d'outils** du canalier, rangées selon les **trois temps de la donnée**.

- **Fabriquer la donnée** (temps 1) : prélèvement et annotation. Comment l'activité du client produit de la donnée juste, captée à l'optimum.
- **Faire circuler la donnée** (temps 2) : fixation et transport. Comment la donnée tient, voyage en local pour piloter, et au loin pour se partager.
- **Lire la donnée** (temps 3) : lecture. Comment un tableau de bord rend praticable la part du lecteur.
- **Entretenir** : maintenance. Comment un canal reste vivant.

Avant ces familles, deux chapitres posent les **méthodes** qui les traversent toutes : l'**architecture hexagonale**, qui rend l'œuvre pérenne en isolant le cœur métier ; et la **gamification de l'expérience**, qui rend l'œuvre agréable et engageante. Après elles, un chapitre rassemble l'**atelier** du canalier — le matériel, les logiciels, les rituels — et un dernier propose des **parcours d'équipement** progressifs.

Chaque chapitre se tient seul. On peut entrer par l'outil dont on a besoin. Mais l'ordre a un sens : on ne lit bien que ce qu'on a su fabriquer et faire circuler.

La règle de ce manuel en une phrase

Les outils changent ; le métier — prélever avec justesse, rendre lisible, rendre autonome — demeure. Ce manuel arme la main pour aujourd'hui ; à la main de savoir qu'elle devra réapprendre demain.

Chapitre 1 — Le canal, le canalier, l'œuvre

Ce chapitre n'apprend rien de neuf à qui a lu la trilogie. Il rassemble, en une langue resserrée, les repères dont chaque outil de ce manuel se sert. On peut y revenir d'un coup d'œil en cours de lecture.

Le fleuve, le canal, la donnée

Il n'y a pas, dans le réel, de données qui attendent qu'on les ramasse. Il y a un **flux du réel** — ce qui se passe, en permanence, indifférent à l'humain, que personne n'a creusé et que personne n'arrête. Prélever, c'est construire un **flux d'observation** : un canal que l'humain détourne du fleuve par son intention, selon des critères qu'il a choisis. Le canal est infiniment plus pauvre que le fleuve ; il ne contient que ce que son tracé a admis. Mais le canal, lui, l'humain le maîtrise : il peut le suivre, le remonter, le rejouer, le relier à d'autres, le confier à un voisin, le combler quand il n'en a plus l'usage.

La **donnée** est le grain d'eau qu'on tire de ce canal. On ne cueille pas une donnée ; on creuse un canal, et la donnée est ce qui s'en écoule.

Pour le solopreneur, le fleuve est son activité qui se déroule : chantiers, courses, devis, appels, livraisons, heures, kilomètres, stocks qui montent et descendent. Tout cela coule, et se perd, si rien ne le capte. Le canalier creuse, dans cette activité, les canaux qui en ramènent ce dont le pilotage a besoin — et rien de plus.

L'intention et les critères

Un canal naît d'une **intention** : suivre un sujet, répondre à une question, piloter une décision. De l'intention découlent les **critères**, qui répondent à cinq questions :

- **Quoi ?** Quels faits bruts de l'activité veut-on prélever ?
- **À quelle cadence ?** À quelle fréquence prélève-t-on ?

- **Avec quelle fidélité ?** Quelle précision, quel instrument, quelle marge d'erreur acceptable ?
- **Avec quel corpus ?** Quelles données de contexte accompagnent chaque prélèvement pour le rendre interprétable (unités, références, date) ?
- **Sur quel périmètre ?** Quelle portion de l'activité observe-t-on, et laquelle laisse-t-on de côté ?

Ces critères ne sont jamais neutres : ils sont la signature de l'intention. Devant tout canal, la première question n'est pas « que dit-il ? » mais « quelle intention l'a fait naître ? ». Un canal ne ment jamais sur le réel ; il ne montre que ce que ses critères lui permettent de montrer.

L'optimum observationnel

Plus n'est pas mieux. Entre deux écueils symétriques — le **sous-prélèvement**, qui passe sous le seuil et perd le signal, et la **sur-ingénierie**, qui prélève bien au-delà du nécessaire et ne gagne rien sinon du coût — il existe une zone où le canal est creusé au mieux. Son point central est l'**optimum observationnel** : le réglage conjoint de la cadence, du corpus et de la probité qui maximise l'écart favorable entre le bénéfice (la qualité du résultat) et le coût (prélèvement, transport, stockage, calcul, temps).

Il n'existe pas d'optimum universel. La question qui le détermine est toujours la même : *l'activité observée peut-elle changer significativement entre deux prélèvements ?* Si oui, la cadence est trop lente. Si non, elle est inutilement rapide. L'optimum se repose à chaque mission. C'est la règle d'or du canalier numérique : ne jamais sur-équiper un client, ni l'aveugler.

Le seuil de Nyquist, en clair. Pour qu'une série de prélèvements restitue fidèlement un mouvement, il faut prélever sensiblement plus vite que la chose ne change. Un stock qui varie chaque jour ne se suit pas par un inventaire annuel. Un chiffre d'affaires mensuel ne se pilote pas à la minute. Accorder la cadence à la vitesse de l'activité : voilà le premier réglage.

Les trois temps, les trois gestes

La donnée a trois temps, posés chacun par un tome. Le canalier numérique les outille tous les trois.

1. **La fabrication** (tome 1) — **l'acte d'inscription** : arrêter une valeur dans un flux qui ne s'arrête pas, et la confier à un support. Trois composantes : *quoi* (l'objet), *quand* (l'instant, la cadence), *comment* (l'instrument, le critère).
2. **La circulation** (tome 2) — **l'acte d'échange** : mettre l'œuvre en mouvement. Le transport n'est pas le dépôt d'une donnée quelque

part ; c'est un rapport entre deux déplacements (la donnée vers le lecteur, le lecteur vers la donnée), accompli seulement quand la part du lecteur est praticable.

3. **La réception** (tome 3) — **l'acte de lecture** : *percevoir* → *distinguer* → *déchiffrer*. Sans lecture, la donnée demeure constituée, mais rien ne s'actualise. Une œuvre livrée que nul ne peut lire est une œuvre inachevée.

L'œuvre du canalier

L'œuvre n'est pas une donnée isolée. C'est un **canal à l'optimum** — creusé, dimensionné, entretenu, restituant fidèlement à l'usage prévu ce qu'il a été conçu pour capter — et, en bout de chaîne, une **donnée consolidée** : la synthèse qui répond à la question d'usage (un indicateur, un statut, une tendance, une alerte). Une œuvre se reconnaît à quatre traits :

- elle est **à la mesure de l'usage** (ni surcalibrée, ni en deçà du seuil) ;
- elle est **durable** dans la limite de ce que sa nature autorise ;
- elle est **signée** — son auteur est nommé, sa date connue, ses critères énoncés ;
- elle est **livrable** — sa part de lecteur est praticable, sa fenêtre de lecteur est nommée.

La posture de référent

C'est l'axe autour duquel tout tourne, et la boussole de chaque choix technique de ce manuel : **aider, rendre autonome, ne pas déposséder**. Trois engagements concrets en découlent, qui jugeront tous les outils du livre.

- **Documentation vivante** : chaque outil s'accompagne d'une cartographie que le client comprend. Il est rédacteur en chef de ses données, non leur locataire.
- **Co-conception** : le client identifie lui-même ce qui mérite l'automatisation et ce qui doit rester manuel ; il garde la souveraineté sur son geste métier.
- **Capacité de lire entretenue** : l'outil rend le dirigeant plus capable de lire ses données, jamais ne l'en dispense.

Critère vérifiable. À tout moment, un client doit pouvoir reprendre la main ou changer de prestataire sans perdre ses données ni sa compréhension. Tout outil de ce manuel qui violerait ce critère est un mauvais outil, si performant soit-il.

La mission récurrente du canalier est celle du forgeron qu'on rappelle pour une pièce nouvelle, non celle du logiciel dont on ne peut plus sortir.

Ce principe, on le verra, commande l'architecture entière de ce qu'on fabrique.

Chapitre 2 — La méthode hexagonale : creuser du pérenne

De tous les outils du manuel, celui-ci n'est pas un logiciel : c'est une **méthode de tracé**. Elle répond à la contrainte la plus dure du canalier numérique — *les outils changent, le métier demeure* — et elle y répond par une discipline d'organisation du code. L'architecture hexagonale (aussi nommée *ports & adapters*) est la traduction technique directe de la posture de référent : elle garantit qu'un client peut changer de technologie sans perdre sa logique métier, et que l'œuvre ne se périmé pas avec la version d'une librairie.

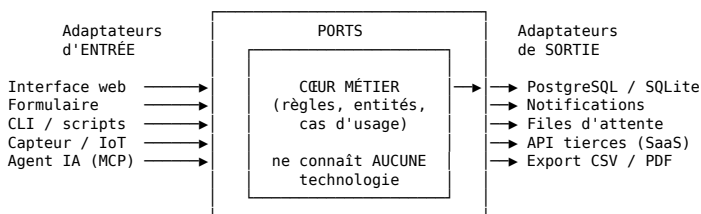
Le principe : isoler le cœur

Une application de gestion mêle deux choses de natures très différentes :

- **Le cœur métier** — les connaissances et les règles propres à l'activité du client. *Un chargement est optimal au-delà de 85 % d'utilisation. Un devis non transformé en commande sous trente jours devient une relance. La TVA bascule au franchissement de tel seuil.* Ce cœur ne dépend d'aucune technologie. Il est le geste métier mis en règles.
- **L'infrastructure** — tout ce par quoi le cœur touche le monde extérieur : la base de données, l'interface utilisateur, les API tierces, les capteurs, les fichiers, les emails. Cette couche est **sujette aux contraintes externes d'évolution** : elle vieillit, se remplace, se reconfigure.

L'erreur ordinaire est de tisser les deux ensemble — d'écrire la règle métier à l'intérieur du code qui parle à la base de données ou qui dessine l'écran. Le jour où la base change, la règle est emportée avec elle.

L'architecture hexagonale **interdit ce mélange**. Elle place le cœur métier au centre, totalement ignorant du monde extérieur, et fait communiquer ce centre avec l'infrastructure par des **contrats** nommés *ports*, dont chaque technologie concrète n'est qu'une implémentation *adaptateur* interchangeable.



À gauche, les **adaptateurs d'entrée** : tout ce qui sollicite le cœur (un clic sur le tableau de bord, un formulaire mobile rempli sur le chantier, un script planifié, un capteur, un agent IA). À droite, les **adaptateurs de sortie** : tout ce que le cœur actionne pour persister ou émettre (enregistrer en base, envoyer une relance, publier dans une file, appeler une API). Le cœur, au centre, ne sait pas si la base est PostgreSQL ou un fichier, ni si l'entrée vient d'un humain ou d'une machine. Il connaît seulement ses *ports*.

Pourquoi c'est l'outil de la pérennité

Trois bénéfices, qui sont autant de promesses tenues au client.

On change la forme sans toucher au fond. Le client démarre avec un stockage local en fichier ; six mois plus tard son volume grandit, on bascule sur une vraie base de données. Côté cœur métier : *rien ne change*. On a seulement remplacé un adaptateur de sortie par un autre, derrière le même port. La règle de gestion qui faisait la valeur de l'œuvre est intacte. C'est exactement la distinction fond/forme de la trilogie, gravée dans le code.

Le métier est lisible et transmissible. La logique métier, isolée, peut s'écrire et se relire en langage clair avant tout code (voir plus loin). Le client peut la valider — « *oui, c'est bien ainsi que je calcule la performance d'un véhicule* » — sans rien connaître à la programmation. C'est la co-conception rendue possible par l'architecture.

On teste le cœur sans rien allumer. Comme le cœur ne dépend d'aucune base ni d'aucun réseau, on peut le vérifier en isolation, avec de faux adaptateurs. Le canal se contrôle avant d'être branché sur le fleuve.

L'infrastructure obéit aux ports

L'ordre habituel du raisonnement est : *j'ai* une base de données, un serveur de mail, un modèle d'IA — et j'écris du code pour m'en servir. L'infrastructure vient d'abord ; le code s'y plie. L'hexagone **renverse** cet ordre. Le port — le contrat — vient d'abord ; l'infrastructure obéit. On ne demande pas « *que sait faire mon infrastructure ?* » mais « *de quoi le cœur a-t-il besoin ?* », et le port l'énonce. Toute technologie concrète

n'est plus alors qu'un adaptateur qui honore ce contrat, interchangeable — y compris dans le temps.

La conséquence va plus loin qu'un simple échange d'adaptateurs. **Un même port peut porter plusieurs adaptateurs, et c'est ce qui est disponible qui décide lequel est monté.** Un port a son adaptateur *dégradé*, quand aucune infrastructure réelle n'est là, et son adaptateur *amplifié*, quand elle l'est. Forme datée, pour fixer les idées : un port « envoi » — sans serveur d'envoi, un adaptateur dépose la demande dans une **file d'attente** que l'artisan reprend à la main ; avec serveur, un adaptateur envoie pour de bon. Un port « consultation d'un modèle » — sans moteur, l'adaptateur répond « *poste requis* » ; avec moteur, il répond. (*Dans l'œuvre de cette édition, on nomme « le poste » cette infrastructure qui honore les ports ; c'est une forme, datée et remplaçable.*) Le cœur, et la règle métier qui appelle le port, **ignorent lequel des deux adaptateurs est branché.**

Le point décisif est là : **l'absence d'infrastructure n'est pas une panne, c'est un cas que le port prévoit.** Quand l'infrastructure manque, le port ne casse pas — il dégrade *fidèlement*. Une demande qui ne peut être honorée maintenant est mise en file (un acte différé, mais tracé) ou renvoie « *il manque ceci* » (une consultation qui dit ce qu'elle attend). Le port tient son contrat même les mains vides : il ne ment pas, il n'échoue pas en silence. Que l'infrastructure obéisse au port, cela veut dire que même son absence est un état prévu, nommé, traversable.

D'où une substitution **locale et totale**. Le jour où une infrastructure réelle arrive, on substitue l'adaptateur — port par port — sans toucher au cœur, ni à la règle métier, ni à l'interface. La fonction qui appelle « *envoyer* » ne change pas selon que l'envoi patiente en file ou part vraiment. C'est le « *on change la forme sans toucher au fond* » déjà vu, étendu : non seulement on remplace une infrastructure par une autre, mais on la voit **apparaître et disparaître** sans que le cœur s'en aperçoive.

Ce renversement a une portée qui dépasse la technique : il dit *qui commande*. L'artisan, gardien du canal, fixe par les ports ce dont le cœur a besoin ; l'infrastructure — une base, un serveur, un modèle, une file — est forme datée : louée, remplaçable, parfois absente, jamais maîtresse. On possède le contrat ; on ne possède pas le monde extérieur, on s'y branche — et s'il n'est pas là, on l'attend dans une file, sans rien rompre.

Les ports dictent l'infrastructure — jamais l'inverse. C'est la maxime à graver au-dessus de l'hexagone : le port est le fond (le besoin durable du métier), l'adaptateur est la forme (la technologie du jour), et la pérennité de l'œuvre tient à ce que jamais le second ne commande au premier.

Le geste du canalier : modéliser avant de coder

Le procédé tient en trois gestes, dans cet ordre, toujours.

1. **Parler le métier du client.** Distinguer le **cœur de métier** — là où l'informatique ne dicte rien, là où réside le savoir-faire du client — de la **matrice de soutien** : les tâches répétitives, administratives, automatisables, qui ne le mobilisent pas (relances, suivi, comptabilité, saisie). On ne touche jamais au cœur de métier ; on modélise la matrice pour libérer le cœur. À la matrice de soutien, commune et générique, répond la **matrice de métier** : l'enchaînement conditionné du travail propre — les états de l'ouvrage et leurs conditions, jamais le geste lui-même. Son principe est exposé au volume jumeau, le *Système de gestion* (section III) ; *Données techniques* dit avec quoi on l'outille.
2. **Écrire la logique métier en clair.** Avant d'ouvrir l'éditeur de code, on rédige les **entités** (les objets du domaine et leurs propriétés), les **règles** (les contraintes), et les **cas d'usage** (les enchaînements). En français, en pseudo-code, en schéma. Ce document se valide avec le client et survivra à tous les langages.
3. **Implémenter dans le langage choisi.** Le développeur transcrit la logique validée. Si le langage change un jour, on retranscrit ; le document de logique, lui, ne bouge pas.

Un exemple : la logique d'optimisation d'un chargement

Voici à quoi ressemble la couche cœur, écrite en clair, indépendante de toute technologie. Elle pourrait être implémentée dans n'importe quel langage sans rien perdre.

Entités du domaine

- *Article* : identifiant, nom, poids unitaire (kg), volume unitaire (m³), quantité, fragile (oui/non), priorité (1 haute → 3 basse), empilable (oui/non).
- *Véhicule* : identifiant, capacité poids (kg), capacité volume (m³), coût au kilomètre, disponibilité.

Règles métier fondamentales

- *RG1 — contraintes physiques.* Le poids total et le volume total chargés ne dépassent jamais les capacités du véhicule ; les deux s'appliquent simultanément.
- *RG2 — fragilité.* Un article fragile majeure de 10 % le volume qu'il occupe (sécurité).
- *RG3 — priorités.* On traite d'abord la priorité haute ; à égalité, on privilégie la forte densité (poids/volume) ; en dernier critère, la grande quantité.
- *RG4 — seuil d'efficacité.* Un chargement est optimal si son utilisation atteint au moins 85 %, l'utilisation étant le minimum entre le taux de remplissage en poids et celui en volume. En deçà, le système alerte.

Cas d'usage — calcul d'efficacité (pseudo-code)

POUR un ensemble d'articles et un véhicule :

```

poids_total    = SOMME(article.poids_unitaire × article.quantité)
volume_nominal = SOMME(article.volume_unitaire × article.quantité)
volume_ajusté  = volume_nominal
POUR chaque article fragile :
    volume_ajusté += article.volume_total × 0,10
efficacité_poids = poids_total / véhicule.capacité_poids
efficacité_volume = volume_ajusté / véhicule.capacité_volume
efficacité = MIN(efficacité_poids, efficacité_volume)
limitation = SI efficacité_poids < efficacité_volume ALORS "poids" SINON "volume"
RETOURNER { efficacité, limitation }

```

Ce texte *est* l'œuvre, au sens du fond. Le code JavaScript ou autre qui l'implémentera n'en est qu'une incarnation datée. C'est cette inversion — la règle d'abord, la technologie ensuite — qui fait l'architecture hexagonale, et qui fait le pérenne.

L'organisation concrète d'un projet hexagonal

Sur le disque, la séparation se lit dans l'arborescence. Le cœur (domain) ne mentionne jamais une technologie ; les ports déclarent les contrats ; les adaptateurs les implémentent ; l'infra héberge les services externalisables.

```

projet-client/
├── domain/           # CŒUR : aucune dépendance technologique
│   ├── entités/      # objets du métier (Article, Véhicule, Devis...)
│   ├── règles/       # règles de gestion (RGL...RGn)
│   └── use-cases/    # enchaînements (optimiserChargement, relancerDevis...)
├── ports/           # CONTRATS
│   ├── input/        # ports d'entrée (commandes, requêtes)
│   └── output/       # ports de sortie (dépôt, notification, publication)
├── adaptateurs/     # IMPLÉMENTATIONS interchangeables
│   ├── storage-sqlite/ # un adaptateur du port "dépôt"
│   ├── storage-postgres/ # un autre, qu'on peut substituer sans toucher au cœur
│   ├── notify-email/
│   └── export-pdf/
├── apps/            # APPLICATIONS clientes (voir ch. 3 et 6)
│   └── web/          # le tableau de bord (HTML/CSS/JS)
└── infra/           # services de base (base, cache, secrets, logs)

```

Garde-fou de sobriété. L'hexagone n'est pas une invitation à multiplier les couches. Pour un solopreneur, le cœur peut tenir en quelques fichiers et un seul adaptateur de stockage. La méthode n'impose pas la lourdeur ; elle impose la *séparation*. On reste à l'optimum : assez de structure pour que le client puisse changer d'outil un jour, pas plus.

Ce que l'hexagone n'isole pas tout seul

L'architecture protège le cœur, mais elle ne dispense pas du jugement. C'est au canalier de décider *ce qui* entre dans le cœur (le savoir métier durable) et *ce qui* reste à la périphérie (les choix datés). Mal tracée, la frontière laisse fuir une règle métier dans un adaptateur, et la promesse de pérennité est rompue. La méthode est une lampe ; le tracé juste reste un geste d'artisan, à reprendre à chaque mission.

Chapitre 3 — Le cœur sans dépendance : HTML, CSS, JavaScript

Si l'architecture hexagonale décide *où* va le code, ce chapitre décide *avec quoi* on écrit le cœur de l'application qui touche le client : son interface et sa logique d'affichage. La réponse du canalier soucieux de pérennité est nette et, à première vue, austère : **le triplet natif du web — HTML, CSS, JavaScript — sans framework ni dépendance**. On parle souvent de *vanilla JS*. Ce choix n'est pas un conservatisme ; c'est une stratégie de durée et de souveraineté.

Pourquoi le natif plutôt qu'un framework

Les frameworks d'interface (React, Angular, Vue et leurs successeurs) rendent des services réels sur les grosses applications d'équipe. Mais pour l'œuvre d'un canalier livrée à un solopreneur, ils introduisent trois fragilités que la posture de référent ne peut pas accepter.

La péremption accélérée. Un framework impose ses versions, ses ruptures, sa chaîne d'outils de compilation. Une application bâtie dessus exige une maintenance permanente *rien que pour rester en place*. Le langage natif du navigateur, lui, ne casse pas : une page HTML/CSS/JS écrite proprement aujourd'hui s'ouvrira encore dans dix ans. C'est la différence entre un canal qu'on doit recreuser chaque saison et un canal maçonné.

Le poids et la dépendance. Un framework ajoute des centaines de kilo-octets et une forêt de dépendances tierces, chacune étant une porte d'entrée pour une faille ou un abandon. Le natif n'ajoute rien : le navigateur est déjà là. L'application reste légère, rapide, compatible avec des appareils modestes — ce qui compte pour un artisan sur un téléphone bas de gamme, dans un camion, hors réseau.

La réversibilité. Une œuvre en natif est lisible par tout développeur, sans connaître un framework particulier. Le client peut changer de prestataire

sans se retrouver prisonnier d'un écosystème. C'est le critère vérifiable du chapitre 1, appliqué à la couche d'interface.

Le natif n'est pas l'absence de méthode. Écrire sans framework ne veut pas dire écrire en vrac. Le cœur applicatif natif s'organise en modules clairs (un module par écran, un module par service), et c'est précisément l'architecture hexagonale qui lui donne sa colonne vertébrale. Le natif fournit la matière ; l'hexagone fournit l'ordre.

Les trois matériaux, et ce que chacun porte

HTML — la structure. Le HTML décrit *ce qui est* : les champs d'un formulaire de saisie sur le chantier, les zones d'un tableau de bord, la liste des devis. Bien écrit (balises sémantiques, libellés explicites, ordre logique), il est lisible par les humains, par les machines, et par les technologies d'assistance. La structure est le squelette de l'œuvre ; elle survit à tout habillage.

CSS — la forme et le ressenti. Le CSS décrit *comment cela apparaît* : couleurs, espacements, lisibilité, hiérarchie visuelle, réactivité à la taille de l'écran (un même tableau de bord doit se lire sur un grand écran d'atelier et sur un téléphone). C'est aussi, on le verra au chapitre suivant, le matériau des animations qui rendent l'expérience agréable — la barre de progression qui se remplit, le *confetti* discret à l'atteinte d'un objectif. Le CSS porte une grande part de la fluidité ressentie, sans une ligne de JavaScript.

JavaScript — le comportement. Le JS décrit *ce qui réagit* : calculer, mettre à jour un indicateur quand une donnée entre, basculer un statut, déclencher une alerte au franchissement d'un seuil. C'est le JS qui anime l'adaptateur d'entrée (la saisie) et l'adaptateur de lecture (l'affichage), et qui appelle le cœur métier. Tenu propre et modulaire, il reste compréhensible et testable.

Application web, application installable : le même cœur

Le même cœur HTML/CSS/JS peut se présenter sous plusieurs formes selon le besoin, sans réécriture du métier :

- **SPA** (*single-page application*) — une application web qui se comporte comme un logiciel, sans rechargement de page. Adaptée au tableau de bord de pilotage consulté au bureau.
- **PWA** (*progressive web app*) — la même application, rendue **installable** sur le téléphone ou l'ordinateur, capable de **fonctionner hors ligne** et de stocker localement. C'est souvent la forme idéale pour le solopreneur mobile : il « installe » l'outil sur son écran d'accueil, saisit un mouvement de stock dans le camion sans réseau, et la synchronisation se fait au retour de connexion.

- **Desktop** via un emballage (type Electron) ou **mobile hybride** via un emballage (type Cordova/Capacitor) — quand le client veut une vraie application installée. Ce ne sont que des **adaptateurs d'entrée** supplémentaires autour du même cœur.

L'important est le principe : *un seul cœur métier, plusieurs habillages d'accès*. On choisit l'habillage à l'optimum du besoin, jamais par défaut de mode.

Le stockage local du cœur

Une application native dispose, dans le navigateur, de réserves locales qui permettent de fonctionner sans serveur :

- des réserves simples pour de petites préférences ;
- une base locale structurée (de type *IndexedDB*) pour conserver des volumes de données métier sur l'appareil, hors ligne, et les synchroniser ensuite.

Pour un grand nombre de solopreneurs, **le local suffit** : leurs données tiennent sur leur appareil, chiffrées, sauvegardées, et ne quittent jamais l'entreprise. Le serveur distant (chapitre 5) n'est mobilisé que lorsque le partage ou la mutualisation l'exigent. C'est la sobriété de l'architecture *local-first* : traiter au plus près de la source, ne faire voyager que ce qui doit voyager.

Plusieurs établis, une seule œuvre

De la convergence des inscriptions séparées. Un artisan peut tenir son œuvre sur plusieurs établis. Ce qu'ils inscrivent séparément ne se contredit pas : cela se succède — chaque inscription demeure un prélèvement figé, un constat daté, porté par l'établi qui l'a produit. Leur réunion n'est donc jamais une copie d'état, où l'un effacerait l'autre : c'est un rejeu d'actes, dans un ordre convenu, où chaque acte demeure. La lecture courante suit l'ordre du rejeu ; l'acte dépassé n'est pas détruit — il rejoint la trace, avec sa date et son établi. Et lorsque deux établis ont inscrit la même chose pendant leur séparation, la divergence est dite à l'artisan : l'œuvre constate, elle ne tranche pas en silence.

Connecter le monde extérieur — toujours par l'hexagone

Le cœur natif ne parle jamais *directement* à un service externe. Tout ce qui vient du dehors — un capteur, une API publique, un service SaaS, un agent d'IA — entre par un **adaptateur** branché sur un **port**. C'est la règle qui rend l'ensemble pérenne et le chapitre 2 en a posé le principe. Quelques exemples de connexions courantes, chacune isolée derrière son port :

- **API publiques de référence** (par exemple les API de l'État pour vérifier un numéro d'entreprise ou normaliser une adresse) : un adaptateur d'entrée enrichit la donnée saisie.
- **Encaissement** (un prestataire de paiement) : un adaptateur de sortie, remplaçable si le client change de prestataire.
- **Logistique, cartographie, messagerie** : autant d'adaptateurs, jamais de dépendances tissées dans le cœur.
- **Capteurs et objets connectés** : un adaptateur d'entrée traduit le signal du capteur en donnée du domaine (voir chapitre 4).

Le jour où l'un de ces services ferme, augmente ses prix ou se périme, on remplace l'adaptateur concerné. Le cœur — le geste métier du client — ne bouge pas. C'est la promesse de souveraineté tenue jusque dans le moindre branchement.

Une note d'honnêteté

Le tout-natif demande de la rigueur : sans framework pour imposer une structure, c'est à l'artisan de la tenir. C'est précisément pourquoi ce manuel insiste tant sur l'architecture hexagonale, sur la modularité, et sur la documentation vivante. Le natif n'est pas le choix de la facilité ; c'est le choix de la durée et de la maîtrise. Pour une œuvre signée, qu'un client doit pouvoir garder et comprendre des années, c'est le bon arbitrage. Comme tout arbitrage, il se repose : un projet d'une tout autre ampleur pourrait justifier d'autres choix — mais ce ne serait plus tout à fait l'atelier, ce serait la chaîne.

Chapitre 4 — La gamification : rendre l'œuvre agréable et engageante

Une œuvre n'est achevée que reçue : si le solopreneur ne lit pas son tableau de bord, ne saisit pas ses mouvements, abandonne l'outil au bout de trois semaines, le canal le mieux creusé ne s'actualise pas, et sa valeur d'usage se perd. La réception n'est pas un supplément ; elle est la condition de l'utilité. La **gamification** — la ludification — est la méthode qui travaille cette réception : elle rend l'usage agréable, motivant, presque sans effort, de sorte que le geste de gestion, d'ordinaire fastidieux, devienne fluide et même satisfaisant.

Le constat qui la justifie est documenté : une part importante des solopreneurs abandonnent leur gestion par **manque de visibilité sur leur performance** et par **désintérêt pour des tâches perçues comme fastidieuses**. La gamification s'attaque exactement à ces deux causes : elle rend la performance *visible* et l'effort *récompensé*.

Le principe : transformer la donnée consolidée en signal motivant

Le tome 1 nomme la donnée consolidée : la synthèse à haute valeur d'usage extraite d'un canal. La gamification est l'art de **présenter cette donnée consolidée sous une forme qui parle au lecteur situé** — qui le renseigne d'un coup d'œil, le situe par rapport à un objectif, et l'encourage. Elle ne fabrique aucune donnée nouvelle ; elle habille la donnée vraie pour qu'elle soit lue.

Le modèle de référence est celui des applications qui ont rendu ludique l'apprentissage ou le suivi d'habitudes : barres de progression, paliers, séries, petites récompenses. Transposé à la gestion d'activité, il donne des mécaniques simples et honnêtes.

Les mécaniques, et le geste métier qu'elles servent

Mécanique	Exemple concret	Ce qu'elle sert
Barre de progression	« Vous avez atteint 80 % de votre objectif mensuel de chiffre d'affaires. »	Visualisation instantanée de l'état (donnée consolidée lisible d'un coup d'œil)
Paliers et badges	« Badge <i>Facturation à jour</i> débloqué après 10 factures émises dans les délais. »	Motivation à compléter les tâches répétitives de la matrice de soutien
Notification ciblée	« La commande n°123 est en retard de livraison — à traiter. »	Réduction des oublis ; alerte au bon moment
Animation de seuil	Un effet visuel discret (<i>confetti</i>) à l'atteinte d'un objectif.	Renforce le sentiment d'accomplissement
Série / régularité	« 7 jours de saisie de stock sans interruption. »	Installe l'habitude de prélever (entretien du canal par le client lui-même)

Toutes ces mécaniques se réalisent en **HTML/CSS/JS natif**, sans aucune dépendance : une barre de progression est une `<div>` dont la largeur suit un indicateur ; un *confetti* est une animation CSS ; une notification est une mise en évidence à l'écran ou une notification système. La gamification n'alourdit pas l'œuvre ; elle la rend vivante avec les matériaux déjà en place.

Les seuils : le pont entre gamification et pilotage

La mécanique la plus utile au solopreneur n'est pas le badge ; c'est le **seuil**. Un seuil est une limite chiffrée qui, franchie, déclenche un signal. Le glossaire du métier en distingue trois sortes, et toutes se gamifient naturellement.

- **Seuil critique bas** — en dessous, l'activité n'est plus pérenne (« *moins de 5 commandes ce mois* »). Signal d'alerte.
- **Seuil critique haut** — au-delà, une restructuration s'impose (« *plus de 20 commandes : envisager de déléguer ou d'embaucher* »). Signal d'anticipation.
- **Seuil intermédiaire** — un objectif que l'entreprise se fixe pour un bénéfice direct (« *15 commandes : une semaine de congés en plus* »). C'est le seuil-récompense, le cœur de la gamification motivante.

Le cas exemplaire est le **franchissement de seuil fiscal** (TVA, plafond de chiffre d'affaires). Pour le solopreneur, ces seuils sont source d'angoisse parce qu'ils sont subis : on les découvre après les avoir dépassés. Une barre de progression vers le plafond, une alerte à 70 % du seuil, transforment l'angoisse en pilotage : *le seuil n'est plus subi, il est anticipé*. C'est, littéralement, rendre au dirigeant la prise sur sa trajectoire — l'objet même de l'œuvre.

Le garde-fou : gamifier sans falsifier

La gamification est puissante, donc dangereuse. Le tome 1 a posé que l'observation insérée dans une boucle décisionnelle **transforme ce qu'elle observe** : un score qui change le comportement qu'il mesure. Mal employée, la gamification peut pousser à de mauvaises décisions — fermer une vente médiocre pour « débloquer un badge », gonfler un chiffre pour faire avancer une barre. Trois règles tiennent ce risque.

1. **Récompenser le geste juste, jamais le chiffre brut.** On gamifie « *facturer dans les délais* », « *relancer un impayé* », « *tenir son stock à jour* » — des gestes vertueux — et non « *faire du chiffre à tout prix* ». La mécanique doit servir la santé de l'activité, pas la flatter.
2. **Ne jamais maquiller une donnée projetée en donnée mesurée.** Une barre qui anticipe une tendance doit se signaler comme projection. Une donnée mesurée et une donnée estimée ne s'affichent pas du même œil. La probité de ce qui est montré prime sur l'effet.
3. **Entretenir la capacité de lire, ne pas l'endormir.** La gamification simplifie la lecture ; elle ne doit pas dispenser de comprendre. Derrière chaque indicateur ludifié, le client doit pouvoir accéder à la donnée brute et à son mode de calcul. L'animation est une porte d'entrée vers la donnée, jamais un rideau devant elle.

Le bon esprit. La gamification n'est pas un divertissement plaqué sur la gestion. C'est une manière de rendre **praticable la part du lecteur** — exactement la définition que le tome 2 donne du transport accompli. Un solopreneur qui n'avait ni le temps ni le goût de lire ses chiffres les lit, parce qu'on les lui a rendus lisibles et engageants. C'est de la littérature de la donnée rendue accessible, pas du vernis.

L'expérience fluide, au-delà des mécaniques

La gamification au sens étroit (badges, barres) s'inscrit dans une exigence plus large : **l'expérience utilisateur**. Pour le solopreneur, fluide veut dire concret :

- **Peu de saisie, bien placée.** Remplacer un cahier oublié par une interface mobile où chaque mouvement met à jour un indicateur ; l'artisan ne « remplit plus un tableau », il dispose d'un inventaire à jour. La saisie se fond dans le geste métier au lieu de s'y ajouter.
- **Autocomplétion et pré-remplissage.** Vérifier un client, normaliser une adresse, reprendre les informations déjà connues : moins le solopreneur tape, plus il reste dans son métier.
- **Lisibilité immédiate.** Un tableau de bord qui répond en un coup d'œil aux trois questions qui comptent (*où en suis-je ? que dois-je faire ? qu'est-ce qui dérive ?*), sans le noyer sous le reste.
- **Sérénité par l'alerte juste.** Être prévenu de ce qui dérive, et *seulement* de ce qui dérive. Une notification qui crie tout le temps

ne dit plus rien ; la sobriété de l'alerte fait sa valeur.

L'expérience fluide, agréable, maîtrisée n'est pas un confort accessoire

c'est le critère de réussite de l'œuvre. Un canal qu'on n'a pas envie de lire est un canal qui ne s'actualisera pas. La gamification, tenue avec probité, est l'outil qui maintient le client *en situation d'observer sa propre activité* — et lui rend la sérénité.

Chapitre 5 — Fabriquer la donnée : prélèvement et annotation

Premier temps de la donnée, premier geste : **l’inscription**. C’est ici que l’activité du client — le fleuve qui s’écoule — devient donnée. Deux familles d’outils servent ce temps : les **outils de prélèvement**, qui matérialisent l’acte d’inscription, et les **outils d’annotation**, qui transforment un signal en donnée qualifiée. Ce chapitre les décline dans leur forme numérique de 2026.

Cartographier la matière : quelles données fabriquer

Avant tout outil, un tri. Toutes les données d’une activité ne se valent pas, et le glossaire du métier les range en trois sortes — distinction qui commande la sécurité, le stockage et le partage.

- **Donnée métier** — directement liée à la réalisation de l’activité : temps passé sur un chantier, matériaux consommés, commandes en cours, kilomètres parcourus. C’est le cœur du prélèvement.
- **Donnée annexe** — collectée indirectement, sans impact direct sur l’activité mais potentiellement valorisable plus tard : positions GPS lors des déplacements, heures d’usage d’une machine (pour une maintenance prédictive future). À prélever avec discernement : utile, mais jamais au prix de la sur-ingénierie.
- **Donnée sensible** — soumise à réglementation (RGPD) ou critique : coordonnées clients, données financières. Elle impose des soins particuliers de stockage et d’accès (chapitre 7).

À ce premier tri se superpose un second axe, qui ne classe plus la donnée par sa nature mais par sa **dépendance**. Certaines données se tiennent seules ; d’autres n’existent que par une autre — une relance par son devis, un document joint par la fiche qu’il documente. La première est dite **portante**, les secondes **portées** (voir, dans les notes complémentaires, *La donnée portée et la donnée portée*). Repérer cette relation à la cartographie évite une faute de modèle : on ne donne pas à une portée une existence autonome qu’elle n’a pas.

La réalisation la plus simple est l'**imbrication** : la portée est logée dans l'enregistrement de sa portante (un champ qui contient ses relances, ses pièces jointes). Elle en hérite mécaniquement le sort — elle est lue, sauvegardée, et **descend en grappe** avec sa portante lorsque celle-ci passe en péremption (voir « Régler la durée limite d'utilisation », plus loin dans ce chapitre). On pourrait tenir la même relation par une table liée avec suppression en cascade : le moyen est substituable, le fait de structure ne l'est pas.

Le prélèvement se modélise sur le **processus métier** du client, décomposé en termes stables : *étape* (phase large : devis, fabrication, livraison), *tâche* (action atomique : découper, envoyer une facture), *séquence opérationnelle* (groupe ordonné de tâches produisant un résultat identifiable : « préparer un chantier »). On prélève là où le pilotage a besoin de voir, pas partout.

Régler la durée limite d'utilisation

Le tome 3 pose le principe : la péremption d'état n'est pas un instant mais un seuil — la **durée limite d'utilisation**, le point où l'écart au réel excède ce que l'usage tolère. Le fond dit qu'un tel seuil existe ; la forme dit ici comment on le règle. Trois cas couvrent l'atelier. **Le seuil intrinsèque** : l'objet de la donnée fixe lui-même le terme (un devis refusé voit son usage clos d'emblée — durée nulle, passage en péremption immédiat ; une offre datée expire à sa date). On l'inscrit comme champ de la donnée, posé à l'acte d'inscription. **Le seuil d'usage** : rien dans l'objet ne fixe le terme, mais l'usage le connaît (un relevé de stock ne sert l'acte qu'une journée ; un prix de fourniture, une saison). On le règle par type de donnée, dans l'armature — c'est un réglage du canal, revu en revue de canal. **Le défaut d'âge** : quand rien n'est spécifié, un défaut prudent s'applique, selon l'âge de la donnée et la cadence de son canal. Au terme, l'œuvre exécute le passage d'état dans l'ordre que le fond impose : la part sommable est consolidée, le nouvel état est écrit, puis seulement la donnée quitte l'usage courant — la péremption ordonne, elle ne détruit pas. Et la grappe suit sa portante : quand celle-ci passe en péremption, ses portées descendent avec elle.

L'armature et le prélèvement : le lit et l'eau

Au tri par la nature (métier, annexe, sensible) et à l'axe de la dépendance (portante, portée) s'ajoute un troisième repère, qui commande la modélisation. Toute donnée n'est pas au même niveau de l'ouvrage : certaines **structurent** — les noms, les types, les formats dans lesquels les autres viennent se ranger (« entreprise », « coordonnée », « type de coordonnée ») ; d'autres sont ce qui **se range** — la valeur d'un SIRET, un numéro composé, un montant. Les premières forment l'**armature** ; les secondes sont les **prélèvements** opérationnels (voir, dans les notes

complémentaires, *L'armature : le canal creusé dans les sources stables*). L'armature est le lit ; le prélèvement est l'eau. Toutes deux sont des données — toute donnée naît d'un prélèvement (voie observationnelle), d'un acte qui institue ou d'une opération qui dérive — mais l'armature est prélevée dans des **sources stables**, le convenu, tandis que l'eau vient du réel volatil de l'activité.

En forme, on ne marque pas cette différence par une convention d'écriture, mais par un **champ**. Chaque nœud du modèle porte un rôle : armature ou prélèvement. Le canalier lit ainsi, d'un coup d'œil, où est le lit et où est l'eau — et l'outil peut traiter différemment ce qui se creuse une fois (l'armature) et ce qui coule sans cesse (les prélèvements).

La liste d'adjacence : tenir l'arbre

L'armature n'est pas plate : une coordonnée porte des téléphones, un téléphone une désignation et un numéro. C'est un **arbre**, et la forme la plus simple pour le tenir est la **liste d'adjacence** : chaque nœud connaît son parent, et l'arbre se reconstitue en suivant les liens. Un nœud, schématiquement :

- id — un identifiant unique horodaté (UUID v7, chapitre 6) : l'identité du nœud ;
- rôle — armature ou prélèvement (le lit ou l'eau) ;
- libellé — le nom porté par le nœud ;
- parent — l'id du nœud dont il relève, ou rien s'il est à la racine ;
- valeur — ce qui est capté, pour un prélèvement (vide pour un nœud d'armature) ;
- references — une liste de renvois (voir plus bas) ;
- et les **métadonnées** que ce chapitre a déjà nommées : date, auteur, unité, contexte.

Cette liste plate est **navigable** (on descend, on remonte, on cherche un nœud par son id) et **extensible** : creuser un canal de plus, c'est ajouter des nœuds, sans toucher aux autres.

Les deux arêtes : la portée et la référence

Deux liens relient les nœuds, et ils ne se réalisent pas de la même façon — c'est le pendant de forme de la note *La donnée portante et la donnée portée* (notes complémentaires), qui distingue la **portée** de la **référence**.

- **Le lien parent réalise la portée** — la composition, le partage de sort. Un nœud relève de son parent ; il naît, vit et **descend en grappe** avec lui. Sa suppression suit celle du parent : **cascade**. C'est l'arête de l'arbre. Une coordonnée sans son entreprise, une désignation sans son téléphone n'ont plus d'objet : on les retire avec.

- **Le champ *references* réalise la référence** — le renvoi vers une donnée à **vie propre**. On y stocke l'id (UUID) de la cible, accompagné d'un **label** qui dit la nature du lien (`{ label: "responsable du poste", cible: <UUID d'une fiche RH> }`). **Pas de cascade** : supprimer la cible ne rompt que le renvoi ; supprimer le nœud qui référence ne touche pas la cible, qui subsiste dans son propre arbre. La référence relie sans nouer l'existence.

Tenir ces deux arêtes distinctes n'est pas un raffinement : c'est ce qui commande, à la suppression, **ce qui meurt en grappe et ce qui survit seul**. Les confondre — mettre un lien de référence en cascade, ou l'inverse — corrompt la sémantique des deux morts.

Imbriquer ou lister : le critère est la recherche

Ce chapitre a déjà donné, pour la portée, une réalisation plus simple encore que la liste : l'**imbrication** — loger la portée dans l'enregistrement de sa portante (un champ qui contient ses relances, ses pièces jointes). Le moyen reste substituable, mais le choix n'est pas indifférent, et un critère le tranche : **la recherche**.

Tant qu'une portée n'est **atteinte qu'à travers sa portante** — les lignes d'un devis, un émetteur figé sur ce devis —, l'imbrication convient : on ne la cherche jamais seule, et la grappe se lit d'un bloc. Mais dès qu'une donnée doit être **cherchée ou parcourue pour elle-même** — retrouver toutes les coordonnées d'un type sur toute la base, lister tous les rendez-vous d'une semaine —, l'imbrication la rend coûteuse à atteindre : il faudrait ouvrir chaque portante pour fouiller son contenu. La **liste d'adjacence** la tient alors mieux : chaque nœud est une entrée à part, indexable, interrogeable directement, sans traverser ses voisins.

La règle est donc : **imbrication pour la portée qu'on ne lit qu'en grappe ; liste d'adjacence dès que la donnée doit se chercher seule**. Choisir l'un pour l'autre ne change pas le fait de structure — la portée reste la portée — mais peut grever le fonctionnement.

L'armature s'entretient

Les nœuds rôle: *armature* — les types, les formats — ne sont pas inventés : ils sont prélevés dans le convenu (les conventions socio-techniques qui fixent comment on désigne une adresse, un numéro, une forme juridique). Ce sont des données de **source stable entretenue** (voir la note *L'enveloppe : la qualité comme entretien du convenu*, dans les notes complémentaires) : durables, mais non éternelles. En forme, cela a une conséquence : l'armature se **versionne** et se **révise**, délibérément, quand la convention se déplace — le jour où une adresse devient un triplet de coordonnées, où le numéro cède à un identifiant. Tenir l'armature

accordée au convenu est un geste d'entretien, de même nature que la revue de canal (chapitre 9), mais portant sur le lit plutôt que sur l'eau.

Garde-fou de forme. Liste d'adjacence, UUID v7, imbrication, champ references : ce sont des moyens datés de 2026. Le fond — l'armature comme canal, la portée et la référence comme deux liens de sorts différents, le convenu comme source stable entretenue — est posé dans les trois notes complémentaires ; ces pages ne font que le réaliser. Un autre support, une autre base, réaliseraient les mêmes faits autrement.

À l'optimum, toujours. Le modèle **supporte** la profondeur ; il ne la **prescrit** pas. Pour une petite activité, on creuse un arbre **peu profond mais complet en canaux** : on exerce les **deux arêtes** — une portée en grappe, une référence vers une autre fonction — sur une armature représentative, de sorte que le potentiel se voie, sans noyer l'atelier sous des prélèvements dont il n'a pas l'usage. Creuser tout serait maçonner un canal de fossé ; c'est le sous-prélèvement et la sur-ingénierie que l'optimum observationnel tient à distance (chapitre 1).

Les outils de prélèvement

Ce sont les instruments qui arrêtent une valeur dans le flux. En version numérique, ils se rangent en quatre groupes.

1. Les formulaires de recueil

L'outil le plus courant et le plus sous-estimé. Un bon formulaire de saisie — un écran mobile rempli sur le chantier, à la livraison, en fin de journée — est un instrument de prélèvement, avec sa cadence et sa fidélité. Les principes :

- **Fondre la saisie dans le geste métier.** L'artisan ne doit pas « remplir un tableau » en plus de son travail ; il enregistre un mouvement au moment où il le fait. Moins de champs, mieux placés, au bon instant.
- **Capter le corpus avec la donnée.** Chaque prélèvement embarque sa date, son auteur, son unité, son contexte — sans quoi il sera ininterprétable plus tard.
- **Pré-remplir et contraindre.** Listes de choix plutôt que texte libre, valeurs par défaut, contrôles de cohérence à la saisie : on empêche l'erreur en amont plutôt que de la corriger en aval.

2. Les scripts d'extraction

Quand la donnée existe déjà dans un autre système (un logiciel comptable, un tableur, une boîte mail, une base existante), on ne la ressaisit

pas : on l'**extraît** par un script. Outils typiques de 2026 : scripts en JavaScript/Node.js ou en shell, lecture de fichiers (CSV, JSON), requêtes sur une base de données existante. Le script d'extraction est un adaptateur d'entrée : il traduit la donnée d'un format étranger vers le domaine du client.

Cas réel. Un directeur de site dispose d'un gros logiciel de gestion mais n'en tire pas les indicateurs dont il a besoin. Le canalier n'installe rien de neuf : il écrit des **requêtes sur la base existante** qui extraient exactement la vue voulue, que le logiciel se contente ensuite d'afficher. Prélever, parfois, c'est savoir interroger ce qui est déjà là.

3. Les capteurs et objets connectés (IoT)

Pour les grandeurs physiques de l'activité — température d'un local, présence, niveau d'un stock via étiquettes (RFID), position d'un véhicule — des **capteurs** prélèvent en continu, à haute cadence. Ils relèvent typiquement de la *donnée annexe* à fort potentiel. Le canalier en connaît la **calibration** (la mesure dérive avec le temps) et les **biais** (ce que le capteur ne voit pas). Le signal brut du capteur entre par un adaptateur d'entrée qui le traduit en donnée du domaine ; il ne s'infiltré jamais dans le cœur métier.

4. L'IA générative comme instrument de prélèvement sémantique

Voici l'instrument le plus neuf et le plus puissant. Un grand modèle de langage (LLM) agit comme un **opérateur sémantique** : il sait lire des informations *non structurées* — un email de client, une photo de bon de commande, un message vocal retranscrit, un document — et en **extraire, qualifier, structurer** la donnée. Là où il fallait jadis un humain pour lire et recopier, ou des dictionnaires et des règles déterministes lourds à maintenir, le LLM prélève le sens directement.

Usages typiques pour le solopreneur : extraction des informations d'une facture reçue (montant, date, fournisseur) ; classement automatique des emails entrants ; qualification d'un avis client ; transformation d'une note manuscrite photographiée en données saisies. La sortie du modèle est **structurée** (en JSON, en tableau) pour entrer dans le système.

Cet instrument se manie selon des règles strictes, détaillées au chapitre 8 : modèle **local** de préférence (souveraineté), **validation humaine** sur tout ce qui est critique, et **combinaison** avec des règles déterministes là où la fiabilité prime. Le LLM ne remplace pas le jugement ; il prépare la matière que le jugement validera.

Les outils d'annotation

Prélever ne suffit pas : il faut **qualifier**. Une grille d'annotation transforme un signal en donnée en lui imposant des catégories — que le flux du réel, de lui-même, ne contient pas. C'est un acte décisif et jamais neutre.

- **Grilles et classifications.** Le canalier construit (ou emprunte) les catégories : types de clients, statuts de commande (préparation, livraison, payée), natures de dépense, niveaux de priorité. Il sait ce que sa grille **inclut et exclut**. Il emprunte une grille existante quand elle est éprouvée et comparable ; il en forge une nouvelle quand l'usage du client exige des distinctions qui n'existent nulle part ailleurs.
- **Protocoles d'annotation.** Des règles stables disent *comment* qualifier, pour qu'un même fait reçoive toujours la même étiquette — condition pour que la série soit comparable dans le temps.
- **Annotation assistée par IA.** Le LLM peut proposer une classification (analyse de sentiment d'un avis, catégorie d'une dépense) ; un humain valide. On combine la rapidité sémantique de la machine et la responsabilité du geste.

L'atelier **conserve ses grilles** dans sa bibliothèque, avec leur histoire et les œuvres qu'elles ont permises. Une grille documentée est un savoir-faire transmissible ; une grille perdue est un canal qu'on ne saura plus relire.

Régler le prélèvement à l'optimum

Tout l'art tient dans le réglage, repris à chaque mission. Pour chaque canal qu'on creuse, le canalier se pose la grille des critères du chapitre 1 :

- **Quoi** : quel fait métier, quelle donnée annexe utile, quelle donnée sensible à protéger.
- **Cadence** : accordée à la vitesse de changement de l'activité. Un stock à mouvement quotidien se prélève à chaque mouvement ; une satisfaction client se mesure à un rythme bien plus lent.
- **Fidélité** : précision de l'instrument, calibration, marge d'erreur acceptable pour l'usage.
- **Corpus** : le contexte embarqué (date, unité, auteur, conditions).
- **Périmètre** : ce qu'on observe et ce qu'on laisse délibérément hors champ.

Et la question qui tranche : *l'activité peut-elle changer significativement entre deux prélèvements* ? Si oui, on densifie ; si non, on allège. Ni aveugle, ni noyé. C'est ce réglage qui distingue l'œuvre du canalier de la collecte indiscriminée — et qui épargne au client le coût et le bruit d'une donnée inutile.

Documenter ses instruments

Un atelier sérieux tient un **inventaire documenté** de ses outils de prélèvement : pour chaque capteur, formulaire, script ou prompt d'extraction, ses caractéristiques, sa calibration, ses biais connus, ses limites. Cet inventaire est la mémoire de la fabrication. Il permet, plus tard, de juger ce qu'une donnée vaut — car juger une donnée, c'est d'abord savoir avec quel instrument elle a été prélevée.

Chapitre 6 — Faire circuler : fixation et transport

Une donnée inscrite et jamais transmise reste solitaire, comme une eau qui ne coule pas. Le deuxième temps de la donnée est celui de la circulation, et il repose sur deux familles d'outils : les **outils de fixation**, qui conservent la donnée sur un support, et les **outils de transport**, qui la mettent en mouvement. Pour le solopreneur, la circulation a deux destinations bien distinctes : **locale**, pour piloter sa propre activité, et **distante**, pour partager. Ce chapitre traite les deux.

Les outils de fixation : où la donnée tient

Le support n'est jamais innocent : il a sa durée de vie, ses contraintes de relecture, ses dépendances techniques. Mal choisi, il expose l'œuvre à la **donnée morte** — le cas où le support survit mais où plus personne ne peut interpréter ce qu'il porte, faute de machine ou de format lisible. Le canalier choisit son support selon la durée d'usage attendue, les contraintes de partage, et le risque de péremption du support lui-même.

Les formats : préférer l'ouvert et le durable

La première règle de fixation est de **ne pas s'enfermer dans un format propriétaire** dont on ne maîtrise pas la pérennité. On privilégie les formats ouverts, lisibles longtemps et par tous :

- **JSON** et **CSV** pour les données structurées d'échange et d'archive : universels, lisibles par une machine comme à l'œil nu, réimportables partout.
- **Markdown** pour la documentation (et, via *pandoc*, la production de PDF) : du texte brut qui survivra à tous les traitements de texte.
- **SQLite** comme base de données dans un simple fichier : une base complète, sans serveur, transportable, ouverte, lisible dans dix ans. C'est souvent le support idéal pour un solopreneur — robuste et sobre.

- **PostgreSQL** quand le volume, la concurrence d'accès ou le partage l'exigent : une base serveur éprouvée et ouverte.

Le canalier **documente le format choisi** pour que les générations suivantes — ou simplement le client, plus tard — puissent encore lire ce qui a été écrit. Un format documenté est une clé de lecture conservée.

L'immuabilité : conserver l'histoire

Pour les données qui engagent (facturation, traçabilité, suivi légal), une bonne pratique de fixation consiste à **ne jamais écraser**, mais à **ajouter**. Plutôt que de modifier un devis en place, on enregistre une nouvelle version et on archive l'ancienne avec son horodatage. Inspirée des registres (*ledgers*) immuables, cette discipline garde une **trace** de tout ce qui s'est passé, ce qui est précieux pour la conformité fiscale et pour rejouer l'historique. Techniquement, on identifie chaque enregistrement par un identifiant unique horodaté (de type *UUID v7*) et l'on peut sceller chaque version par une empreinte (un *hash* de son contenu), de sorte que toute altération se détecte.

À l'optimum, toujours. L'immuabilité a un coût (volume, complexité). On la réserve à ce qui la mérite — les données engageantes, traçables, durables — et l'on garde un stockage simple pour le reste. Inutile de maçonner un canal de fossé.

Le système d'événements : faire circuler l'acte d'inscription

Dans une œuvre vivante, chaque geste — créer un devis, relancer un client, clôturer une échéance — est un **acte d'inscription** : un prélèvement sur le flux du réel, fixé. Pour qu'une œuvre soit davantage que la somme de ses fonctions, ces inscriptions doivent **circuler** entre les modules sans qu'aucun ne dépende d'un autre. C'est le rôle d'un système d'événements.

L'**événement** est la forme logicielle de l'acte d'inscription. On lui donne un nom — fonction.entité.action —, une date, une source, et les données du fait. Il est, au sens où la trilogie l'entendait, un **datum** : « *ce qui est donné — pour une occasion ultérieure* ». Un module émet l'événement sans savoir qui l'écoute ; d'autres s'y abonnent pour enclencher, à partir de lui, une action nouvelle — manuelle, automatisée ou déléguée. Une même inscription peut ouvrir plusieurs occasions.

Deux organes le portent. Le **bus** transporte l'inscription, ici et maintenant, des émetteurs vers les abonnés : c'est le transport local, **découplé**, qui permet de greffer une fonction sans toucher aux autres. Le **journal** conserve la succession des inscriptions : c'est le flux d'observation rendu durable, immuable, ajouté en fin (*append-only*) — la **source fiable** de l'activité. L'état courant d'une fiche n'est qu'une lecture commode ; le journal est la trace. On y rejoue, on y consolide, on y audite.

Chaque acte s'ouvre et se ferme par une inscription. Avant d'agir, on inscrit l'intention (init) avec ses paramètres : si l'acte échoue durement, on remonte à son origine. L'acte accompli, on inscrit sa clôture (finish) ; l'acte échoué de façon contrôlée, son échec (fail) — qu'un module peut écouter pour alerter aussitôt. La boucle de l'acte est elle-même tracée.

Garde-fou de forme. Tout ceci est daté : un bus en mémoire, un journal local, pour une œuvre mono-poste. Le fond demeure l'ontologie — l'acte d'inscription, le datum, le flux d'observation. Demain, le bus pourra être distribué et le journal archivé ailleurs ; l'inscription, elle, reste ce qu'elle est : un fait donné pour une occasion ultérieure.

L'archivage et les obligations

Certaines données doivent être conservées par obligation légale (des factures pendant plusieurs années). L'**archivage** est la conservation longue durée de données inactives mais utiles ou obligatoires. Il se distingue du stockage courant : on peut le compresser, le chiffrer, le ranger à part, voire programmer sa **suppression automatique** au terme légal (par exemple un effacement après la durée de rétention prévue, par souci de conformité RGPD). Le canalier inscrit ces règles dès la conception.

Le transport local : piloter son activité

La circulation la plus quotidienne pour le solopreneur ne va nulle part au loin : elle relie ses propres prélèvements à son propre tableau de bord. C'est le transport **interne**, qui transforme une activité dispersée en pilotage lisible.

Architecturalement, c'est le mouvement *du cœur métier vers la lecture* : la donnée fabriquée (chapitre 5) est consolidée — moyenne, total, statut, tendance — puis acheminée vers l'écran qui la rendra visible (chapitre 7). En local-first, tout cela se passe **sur l'appareil du client**, sans serveur distant : la donnée brute ne quitte pas l'entreprise, le pilotage est instantané, et il fonctionne hors ligne.

Le mécanisme typique : chaque mouvement saisi met à jour un indicateur d'état ; le tableau de bord lit l'indicateur consolidé, jamais la masse brute. Le solopreneur ne « consulte pas un journal de toutes ses opérations » — il voit *où il en est*. C'est la consolidation (tome 1) au service de la fenêtre du lecteur (tome 3).

Quand plusieurs appareils ou plusieurs applications du même client doivent se coordonner sur une même machine ou un même réseau local, des outils de circulation interne entrent en jeu : une base partagée, un cache rapide (de type Redis) pour les états fréquemment lus, une file de messages (de type RabbitMQ) pour faire passer des événements d'un service à l'autre sans les coupler, des tâches planifiées (*cron*) pour les

consolidations périodiques. Pour un solopreneur, la plupart du temps, **rien de tout cela n'est nécessaire** : un fichier de base et un peu de JavaScript suffisent. On ne mobilise ces outils qu'au franchissement réel d'un seuil de complexité — à l'optimum, encore.

Le transport distant : partager la donnée

Le second usage de la circulation porte vers l'extérieur : transmettre à un tiers (l'expert-comptable, un client, un partenaire), ou faire entrer sa donnée dans un échange plus large. Le tome 2 donne la **typologie minimale** du transport, et chaque mode appelle ses outils.

- **Déplacer** — porter la donnée d'un lieu à un autre (une sauvegarde, une migration). Outils : export de fichiers, copie chiffrée.
- **Transmettre** — la faire passer à un destinataire désigné (envoyer le récapitulatif annuel au comptable). Outils : export **CSV/PDF**, message, dépôt sécurisé.
- **Rendre accessible** — la mettre à disposition de qui viendra l'interroger. Outils : une **API** (interface programmable) avec authentification.

Le transport n'est accompli que si la part du lecteur est praticable

C'est la règle d'or, héritée du tome 2 : transmettre n'est pas déposer. Livrer au comptable un export brut qu'il devra retraiter, ce n'est pas transmettre — c'est tomber. Le canalier **consolide** et **met au format attendu par le destinataire** pour que celui-ci puisse accomplir sa part sans effort. Pour le solopreneur qui partage, cela veut dire : un export propre, daté, dans le format que le tiers sait ouvrir, accompagné de ce qu'il faut pour l'interpréter.

L'API et le protocole MCP : ouvrir sa donnée aux agents IA

Un besoin neuf de 2026 : rendre sa donnée interrogeable par des **agents d'IA**. De plus en plus de clients veulent qu'une IA puisse lire leur agenda, leurs commandes, leur activité, pour les assister. La réponse technique est de développer une **API client** — des points d'accès qui exposent, de façon contrôlée et authentifiée, les données voulues — et, pour qu'elle dialogue avec les modèles d'IA selon un standard, de la rendre conforme au protocole **MCP** (*Model Context Protocol*), qui normalise l'interconnexion entre une source de données et un agent.

Côté architecture, cette API n'est qu'un **adaptateur** de plus, en entrée et en sortie de l'hexagone : on l'ajoute sans toucher au cœur métier. C'est l'illustration parfaite de la pérennité par l'architecture : un logiciel vieux de vingt ans, dont les données étaient prisonnières de son interface,

retrouve une seconde vie en exposant une API MCP — sans réécrire sa logique.

Les altérations du transport, à prévenir

Le transport n'est pas neutre : il peut perdre des unités, des métadonnées, du contexte ; il prend du temps, pendant lequel la donnée vieillit. Le canalier connaît ces altérations et les prévient : il embarque le corpus dans l'export, il horodate, il signale la fraîcheur, il vérifie qu'une donnée fraîche à la source n'arrive pas périmée. Un atelier documente, pour chaque type d'œuvre, le **mode de transport éprouvé** et ses conditions de réception.

La multiplication : un fait partagé n'est plus un objet unique

Particularité de la donnée : la faire voyager peut en **multiplier les porteurs** (une copie ailleurs sans cesser d'exister ici). Cela change la manière de raisonner sur la fiabilité, la propriété, la sécurité. On ne sécurise pas un fait diffusé en de multiples lieux comme on garde un exemplaire unique. Avant de partager, le canalier — et le client — se demandent : *une fois sortie, cette donnée ne reviendra pas ; qui pourra en faire quoi ?* La circulation distante engage ; elle se décide en conscience, pas par défaut.

Réutiliser sans transmettre : le moment objectal, et comment le fond le tient

La typologie du transport laisse un cas de côté : celui où le canalier **réutilise**, à l'intérieur de l'œuvre, une donnée qu'il détient déjà — sans l'envoyer à personne. Le devis qui reprend la fiche d'un client en est l'exemple : la consolidée passe d'un support à un autre — de la fiche au devis —, mais elle reste **chez le même destinataire**, l'artisan. C'est une navigation spatiale (tome 2), mais *interne* et *sans destinataire tiers* : aucun porteur supplémentaire n'est créé vers l'extérieur, la donnée n'est pas cédée, pas même partagée. On réutilise ; on ne transmet pas.

Ce geste mérite d'être nommé, car il touche l'ontologie. Réutiliser une consolidée comme un repère stable qu'on désigne, qu'on copie, qu'on rattache, *c'est la traiter, le temps d'un usage, comme un objet*. Les **quatre présupposés** que le tome 1 prête à l'ontologie objectale s'y vérifient localement : la consolidée semble **préexister** au devis, on la tient pour **stable**, on lui reconnaît une **valeur d'usage**, on la **possède** et on la manipule. C'est le **moment objectal** de l'œuvre.

Faut-il s'en alarmer ? Non — et c'est tout le point. Le tome 1 l'a déjà dit : le *datum* y est « ce qui peut être **repris** dans l'occasion suivante », un **résidu**

de processus. Reprendre une consolidée ne trahit pas le processuel : c'en est le mécanisme même. La donnée consolidée est, *par définition* (tome 1), « réutilisable dans d'autres contextes ». Le moment objectal n'est donc pas une rechute dans l'ontologie dominante : c'est une **forme datée** que le **fond** processuel prévoit, autorise — et **borne**.

Trois bornes, que l'œuvre rend visibles là où l'ontologie objectale ordinaire les oublie :

- **La péremption.** La consolidée réutilisée est un résidu détaché de son flux d'observation. Le nom du client, figé sur le devis, ne suit plus la fiche vivante. Quand la fiche disparaît, l'œuvre l'**affiche** — un *lien rompu* — au lieu de feindre que le repère vaut encore. La péremption est montrée, jamais masquée.
- **L'inappropriabilité du flux.** On possède le **prélèvement** et l'**annotation** — le devis, sa consolidation —, jamais le flux d'où ils sont tirés (quatrième renversement, tome 1). Supprimer le contact ne casse pas le devis : on n'a jamais possédé que le résidu.
- **La recontextualisation.** Le *datum* est **relationnel** (tome 1) : l'identifiant repris ne signifie « ce client » que *dans ce devis*, comme une relation neuve — non comme une substance transportée intacte.

La règle de pratique tient en une phrase : *réutiliser le résidu sans jamais oublier qu'il est résidu*. Ce qui sépare l'œuvre de l'approche dominante — les *data lakes*, « mes données » tenues pour un bien pérenne — n'est pas qu'elle s'interdise de manipuler la consolidée comme un objet ; c'est qu'elle ne perd jamais de vue qu'elle se périme, qu'elle ne possède pas le flux, qu'elle recontextualise au lieu de transporter du sens tout fait. Le moment objectal est légitime *tant que le fond le tient*.

Garde-fou de forme. Tout cela est daté : un identifiant, un nom figé, un marqueur de lien rompu. Le fond demeure : *on réutilise un résidu, on ne possède pas un flux ; manipuler la donnée comme un objet est permis le temps d'une occasion, à la condition de se souvenir qu'elle se périme*. C'est ici, au point de réutilisation, que l'ontologie objectale opère dans l'œuvre — et c'est ici, exactement, que le fond processuel doit la tenir.

Le flux entrant : prélever le courrier

Jusqu'ici, le flux du réel que l'œuvre observait était celui de l'activité de l'atelier : un devis posé, une relance, une facture. Mais un autre flux arrive, du dehors : le courrier. Un message reçu n'est pas un objet rangé dans une boîte ; c'est, lui aussi, un flux du réel — une parole adressée, datée, qui ne se répétera pas à l'identique. L'œuvre n'en fait pas une exception : elle le traite comme tout flux, par un **prélèvement**.

Ce prélèvement a une exigence propre, que la forme datée rend visible. Le courrier ne vit pas dans l'œuvre : il vit sur une source, au-dehors, que l'œuvre ne possède pas. Prélever, ici, c'est **lire sans altérer la source**. Le poste relève les messages en lecture seule : il ne marque rien, ne déplace

rien, ne consomme rien sur le serveur ; il en prend une trace et laisse le flux intact. C'est l'**inappropriabilité du flux** reconduite à l'entrée : on ne s'empare pas du courrier, on en prélève une inscription. La source reste celle d'un autre ; l'œuvre n'en garde qu'un dépôt, daté, qui lui appartient en propre.

Une fois prélevé, le message est **inscrit**. Il entre dans les deux temps de la donnée comme n'importe quelle observation : un premier temps, le message tel qu'il fut reçu — l'expéditeur, le sujet, le corps, intacts ; puis un second temps, où il est repris dans l'occasion suivante — lu, classé, répondu. L'œuvre tient la trace de ce parcours : ce message est-il neuf, lu, traité ? La boîte du dehors l'ignore ; l'œuvre, elle, le sait, parce qu'elle a inscrit non pas le courrier, mais *sa lecture par l'artisan*.

Reste la lecture elle-même. Présenté au **lecteur situé**, le message appelle l'acte de lecture : percevoir qu'il est là, le distinguer des autres, le déchiffrer. Sur ce dernier geste, le déchiffrement, un appui est possible — et c'est ici qu'il faut être précis sur ce que l'appui est, et n'est pas. Une consultation peut proposer une **qualification** (de quel ordre est ce message ?) et un **brouillon** de réponse. Ces propositions ne sont pas le message : ce sont des données **vraisemblables**, non des données observées. Le message reçu est observé ; ce qu'on en infère est vraisemblable, et doit le rester — affiché comme tel, *à valider*. L'artisan relit, ajuste, décide. Le brouillon n'a d'effet que par son acte ; rien ne part sans qu'il l'ait fait sien. L'appui déchiffre une première fois ; il ne lit pas à la place du lecteur. Le point où le réel se relit reste l'humain.

Tout cela tient au fond, et le passage par le courrier électronique n'en est que la forme. Que le flux entrant soit relevé par tel protocole aujourd'hui, par tel autre demain, ne change rien à ce qui est dit : un flux du réel arrive du dehors ; on le prélève sans l'altérer ; on l'inscrit avec sa trace ; un lecteur situé le déchiffre, assisté mais non remplacé. La voie est datée. Le geste, lui, est celui de toujours.

Chapitre 7 — Lire la donnée : le tableau de bord de pilotage

Troisième temps, troisième geste : **la lecture**. C'est là que la donnée devient utile, ou demeure non lue. Le tome 3 l'a établi : *sans lecture, rien ne s'actualise*. L'outil-roi de ce temps, pour le solopreneur, est le **tableau de bord de pilotage**. Ce chapitre dit comment le concevoir pour qu'il soit réellement lu — c'est-à-dire pour qu'il respecte la **fenêtre du lecteur**.

Le lecteur est situé, sa fenêtre est finie

Aucune lecture n'est de nulle part. Le solopreneur lit son activité depuis une situation précise : peu de temps, une attention déjà sollicitée, un corpus de compétences donné, souvent un petit écran entre deux tâches. Sa **fenêtre de lecteur** est étroite. Le canal le mieux creusé, transporté avec la plus grande fidélité, ne sert à rien si ce qu'il livre dépasse cette fenêtre. *Une donnée qu'on n'a pas la capacité d'examiner n'est pas lue, quoi qu'on en croie.*

La conception d'un tableau de bord est donc, avant tout, un travail sur la fenêtre : ne montrer que ce qui peut être lu, dans le temps et avec l'attention dont le client dispose. C'est l'exact pendant, côté réception, de l'optimum observationnel côté fabrication : ni aveugle, ni noyé.

La sélectivité : moins, mais juste

La lecture est **sélective** par nature : on ne lit pas tout, on choisit. Un bon tableau de bord choisit *à la place du lecteur* ce qui mérite son attention rare. Il répond, d'un coup d'œil, à trois questions et à elles seules :

1. **Où en suis-je ?** L'état courant, consolidé en signaux simples : chiffre d'affaires du mois contre objectif, trésorerie, commandes en cours, marge.
2. **Que dois-je faire ?** Les actions qui appellent une décision : un devis à relancer, une facture en retard, un stock sous le seuil.

3. **Qu'est-ce qui dérive ?** Les alertes : un seuil approché, une tendance qui se retourne, une anomalie.

Tout le reste — la masse des prélèvements bruts — reste accessible *en profondeur*, mais ne s'affiche pas en surface. Le tableau de bord livre la **donnée consolidée**, pas le journal. La règle de Lavoisier du canulier : *consolider l'activité en signaux simples, plutôt que livrer une masse brute indéchiffrable*.

Les indicateurs : choisir les bons

Un **indicateur** est une valeur calculée à partir des données, destinée au pilotage et à la décision. Le canulier en distingue les types et les choisit avec le client, jamais à sa place :

- **Indicateurs d'état** — mesurent l'instant : nombre de commandes en retard, niveau de stock, trésorerie.
- **Indicateurs de tendance** — mesurent le mouvement : évolution du chiffre d'affaires, taux de conversion devis → commande.
- **Indicateurs de seuil** — situent par rapport à une limite : progression vers le plafond de chiffre d'affaires, vers un objectif (voir la gamification, chapitre 4).

Pour les activités où le pilotage financier est trop volatil ou trop tardif, on peut s'appuyer sur des **indicateurs physiques** plutôt que monétaires — heures de travail, volume produit, kilomètres, taux d'utilisation d'un véhicule, énergie consommée — qui décrivent l'activité de manière stable et immédiate, la traduction financière se faisant après coup. Cette approche par le flux physique donne un pilotage robuste, peu sensible aux aléas, et souvent plus parlant pour un artisan que des écritures comptables.

Cas réel. Une entreprise de transport perd de l'argent sur des livraisons mal remplies. L'indicateur juste n'est pas global : c'est la **performance unitaire de chaque véhicule selon son type de chargement** (taux d'utilisation, coût/bénéfice). Bien choisi, l'indicateur transforme une intuition floue en décision argumentée : quels moyens maintenir, lesquels réduire. Choisir l'indicateur, c'est déjà la moitié de l'œuvre.

La fraîcheur de la lecture : irriguer le tableau de bord

Une lecture se prend à un instant ; le flux, lui, ne s'arrête pas. Un tableau de bord consulté puis laissé tel quel perd vite sa **fraîcheur** : il montre un état révolu du flux d'observation pendant que l'activité, elle, a continué. Le lecteur situé ne devrait ni se demander si ce qu'il voit est à jour, ni recharger la page à la main pour s'en assurer. *Une lecture qui retarde sur le flux ne lit plus tout à fait le présent : elle lit un état révolu.*

Le principe, côté réception, est simple : la surface de lecture doit être **irriguée** par le flux. Ce qui s'inscrit apparaît, là où le lecteur regarde. Le tableau de bord n'est pas un cliché qu'on rafraîchit de mémoire ; c'est une surface tenue à la fraîcheur du flux d'observation. C'est le pendant exact, au troisième temps, de l'optimum observationnel du premier : ni figé, ni clignotant.

Concrètement — et de façon datée — chaque acte s'inscrit comme un événement ; la surface écoute le flux et se redessine quand il grandit. Trois garde-fous tiennent cette irrigation dans la **fenêtre du lecteur**. D'abord, *ne rafraîchir que ce qui est sous les yeux* : la fenêtre est finie, inutile de recalculer une vue qu'on ne regarde pas. Ensuite, *ne jamais écraser un geste en cours* : le lecteur est aussi un acteur ; si sa main est dans un champ, l'irrigation attend — elle ne reprend pas ce qu'il est en train d'inscrire. Enfin, *regrouper les rafales* : un rafraîchissement par cycle, non un par événement, sans quoi le scintillement déborderait la fenêtre aussi sûrement qu'une masse brute.

Deux compléments achèvent le dispositif : un « **rafraîchir** » à portée de main, pour le lecteur qui veut forcer la mise à jour, et un rafraîchissement au **changement de vue**, le temps ayant pu passer. Ainsi le tableau de bord reste à l'optimum : ni en retard sur le flux — préemption silencieuse de la lecture —, ni débordant.

Garde-fou de forme. Tout ceci est daté : un bus en mémoire, un redessin local. Le fond demeure : *la lecture doit rester fraîche par rapport au flux*. Les moyens de l'irrigation pourront changer ; l'exigence de fraîcheur, elle, ne se périmé pas.

Les outils de lecture, et leurs biais

Les outils qui donnent à voir — graphiques, visualisations, requêtes, croisements, statistiques descriptives — **transforment ce qu'ils lisent**. Une visualisation choisit ses axes, ses échelles, ses agrégations ; elle peut révéler une tendance réelle ou en suggérer une fausse. Le canalier connaît ces biais comme il connaît ceux de ses capteurs, et il en informe le client. Concrètement, pour le solopreneur, on s'en tient à des représentations honnêtes et lisibles :

- des **chiffres-clés** mis en avant (le strict nécessaire) ;
- des **courbes** simples pour les tendances, à échelle non trompeuse ;
- des **barres de progression** vers les objectifs et les seuils ;
- des **listes d'actions** triées par urgence ;
- un **code couleur** sobre (par exemple : vert/orange/rouge) pour l'état et l'alerte.

Tout cela se réalise en **HTML/CSS/JS natif** (chapitre 3) : pas besoin de bibliothèque lourde pour dessiner une barre, un chiffre, une courbe simple. L'outil de lecture reste léger, rapide, ouvrable partout, et réversible.

Les trois postures du lecteur, et ce qu'elles dictent à l'interface

Le tome 3 décrit trois manières d'habiter le sur-débit. Chacune se traduit en choix d'interface.

- **Cadrer** — restreindre ou déplacer sa fenêtre pour lire vraiment ce qui importe. À *l'interface* : laisser le client choisir ses indicateurs, masquer le reste, aller à l'essentiel. Le tableau de bord est un cadrage outillé.
- **Déléguer** — confier la lecture à un dispositif (un automatisme, un agent IA) dont on **héríte la situation**. À *l'interface* : permettre la délégation (un résumé automatique, une alerte intelligente) tout en signalant *qui* a lu et avec quelles limites. On délègue la lecture, pas la responsabilité.
- **Accepter** — consentir avec lucidité à ne pas tout lire. À *l'interface* : la sobriété. Ne pas culpabiliser le client de ne pas tout voir ; certaines données à fraîcheur brève n'avaient pas vocation à être lues, et ce n'est pas un échec.

La pire posture, non écrite, est la quatrième : *croire qu'on a lu ce qu'on n'a fait que voir passer*. Un tableau de bord qui donne l'illusion de la maîtrise sans la donner réellement est nuisible. La probité de l'outil consiste à rendre lisible *ce qui est*, et à donner accès, sous la surface, à la donnée et à son mode de calcul.

Entretenir la capacité de lire

Garde-fou répété parce qu'il est central : l'outil de lecture rend le dirigeant **plus** capable de lire ses données, jamais ne l'en dispense. Chaque indicateur affiché peut être « ouvert » : d'où vient-il, sur quelle période, calculé comment, avec quelle marge. Déléguer à l'interface la fatigue du calcul est un bon arbitrage ; déléguer la compréhension est un mauvais. Le tableau de bord est une lampe que le client apprend à tenir, pas un oracle qui pense à sa place.

La fenêtre nommée, dans la signature

Toute œuvre de lecture se livre en nommant la **fenêtre de lecteur** qu'elle vise : à qui ce tableau de bord est destiné, dans quelles conditions il est lisible, ce qu'il faut savoir pour l'interpréter. Ce champ figure dans la fiche de signature (chapitre 9). Concevoir pour un lecteur situé, et le dire, fait partie de la livraison — c'est ce qui distingue une œuvre achevée d'une donnée jetée à l'écran.

Chapitre 8 — L'IA locale : l'opérateur sémantique du canalier

L'IA générative est, pour le canalier de 2026, l'outil le plus puissant et le plus délicat. Elle traverse les trois temps : elle prélève du sens (fabrication), elle aide à transmettre et résumer (circulation), elle lit à la place du client (réception). Le tome 1 l'a située avec précision : le grand modèle de langage condense en une entité les trois rôles historiquement séparés — la **bibliothèque** où l'information est rangée, le **bibliothécaire** qui la retrouve, le **conteur** qui la restitue. C'est un **saut systémique**, avec ses bénéfices et sa contrepartie. Ce chapitre dit comment l'employer en artisan : localement, avec mesure, et sans déposséder le client.

Ce que l'IA générative sait faire pour le solopreneur

Comme **opérateur sémantique**, un LLM traite des informations non structurées sans qu'on ait à maintenir des dictionnaires ni des règles complexes. Les tâches de gestion qu'il automatise, en tout ou en partie :

- **Qualifier et classer** — trier des emails, catégoriser des dépenses, analyser un avis client.
- **Extraire et structurer** — tirer d'une facture reçue ou d'un bon de commande les champs utiles, en sortie **JSON** exploitable.
- **Générer** — rédiger un brouillon de devis, de relance, de réponse type, un compte rendu.
- **Synthétiser** — résumer un dossier, une journée d'activité, une série d'échanges.

Le principe directeur, partout : **l'IA propose, l'humain décide**. Pour les tâches répétitives, structurées et à faible risque, l'automatisation peut être complète. Pour tout ce qui est critique, sensible ou juridique — la validation d'un devis, d'une facture, d'un contrat, d'une fiche de paie — **la décision et la validation restent humaines**, sans exception.

Pourquoi local : souveraineté, coût, maîtrise

Le canalier privilégie l'**IA locale** — un modèle qui tourne sur la machine du client ou de l'atelier, et non dans un cloud distant. Trois raisons, qui sont les trois piliers du métier :

- **Souveraineté et confidentialité.** La donnée — souvent sensible — **n'a pas à quitter l'entreprise**. C'est l'architecture *edge / local-first* appliquée à l'IA : traiter au plus près de la source. La surface d'attaque diminue ; la conformité s'en trouve simplifiée.
- **Maîtrise des coûts.** Pas d'abonnement par requête, pas de facture qui croît avec l'usage. Le coût est celui du matériel, amorti.
- **Pérennité et indépendance.** On ne dépend pas de la politique d'un fournisseur distant qui peut changer ses prix, ses conditions, ou fermer.

Les outils de 2026 rendent cela accessible : des **modèles ouverts et légers** (de la famille des Llama, Phi, Gemma et autres), capables de tourner sur un ordinateur portable correct ou un petit serveur ; et des **runtimes locaux** (de type *Ollama*) qui les installent et les servent en quelques commandes. On choisit le modèle au juste besoin : un petit modèle pour classer des emails, un modèle un peu plus capable pour de l'extraction ou de la rédaction. Inutile de viser le plus gros : à l'optimum, encore.

Quand le cloud reste légitime. L'IA locale est la règle, pas un dogme. Une tâche lourde et ponctuelle, ou un besoin que le matériel local ne couvre pas, peut justifier un recours mesuré à un service distant — à condition de le dire au client, de ne pas y faire transiter de donnée sensible sans son accord éclairé, et de l'isoler derrière un adaptateur remplaçable. L'arbitrage se pose, comme toujours.

Combiner l'IA et les règles déterministes

L'IA n'est pas toujours le bon outil. Pour les tâches **structurées, déterministes, où la fiabilité prime** — un calcul de TVA, l'application d'une règle de gestion, une validation de format — les méthodes classiques (règles explicites, automatisation de tâches répétitives de type RPA, dictionnaires) restent supérieures : prévisibles, vérifiables, sans hallucination. Le canalier **combine** : l'IA pour le sémantique et le flou, la règle déterministe pour l'exact et le critique. Souvent, la sortie d'un LLM est *contrôlée* par une règle déterministe avant d'être acceptée — indicateurs, seuils, contrôles de cohérence détectent l'anomalie.

La méthode d'automatisation progressive

Équiper un solopreneur d'IA ne se fait pas d'un bloc. La méthode est **progressive et collaborative** : commencer petit, valider, étendre. Six étapes.

1. **Audit initial.** Par un entretien guidé, recueillir les tâches quotidiennes, hebdomadaires, mensuelles. Les classer selon quatre critères — temps consommé, pénibilité, risque d'erreur, impact sur la qualité — et les **prioriser** dans une matrice. Livrable : une liste priorisée des tâches candidates, validée par l'entrepreneur. On cible d'abord ce qui est chronophage, pénible et source d'erreurs.
2. **Cartographie des tâches automatisables.** Croiser la liste avec la faisabilité (données disponibles, matériel, compétences) et proposer, pour chaque tâche prioritaire, la solution (modèle adapté, outils complémentaires, méthode de validation). Livrable : un tableau par tâche — solution, matériel requis, bénéfices, risques.
3. **Prototypage et test.** Sur **une ou deux** tâches simples et impactantes (classer les emails, générer des brouillons de réponse) : installer les outils, configurer, **former l'entrepreneur**, puis tester deux à quatre semaines en mesurant gains de temps, erreurs, ressenti. Livrable : un rapport de test.
4. **Déploiement progressif.** Par vagues — deux nouvelles tâches tous les un à deux mois. Cibles fréquentes : gestion des emails, suivi client/fournisseur, stocks et tarifs, création de documents, veille. Insister sur la **validation manuelle** et la compréhension des limites. Livrable : un tableau de bord de suivi.
5. **Optimisation.** Affiner les prompts, ajuster les règles, automatiser des tâches plus complexes (analyse des ventes, rapports mensuels), faire évoluer le matériel si besoin. **Encourager l'autonomie** : le client sait modifier des règles simples, la documentation est claire.
6. **Maintenance.** Mettre à jour les modèles, sécuriser, sauvegarder, faire des points réguliers. Livrable : des modalités de suivi claires.

Le verbatim qui guide la relation au client résume l'esprit : « *L'IA est là pour vous aider, pas pour vous remplacer.* » — « *Commencez petit, voyez grand.* » — « *Vous décidez ce qui est automatisé et comment.* »

La contrepartie, nommée

Le canalier dit la vérité, y compris ce qu'un vendeur tairait. Déléguer le traitement d'une donnée à une IA, c'est **s'en éloigner un peu**. Le tome 1 l'a posé : la fusion des trois rôles (bibliothèque, bibliothécaire, conteur) fait disparaître ce que leur séparation rendait possible — la traçabilité (savoir d'où vient une information), le retour à la source, la critique, le contre-pouvoir d'un rôle sur l'autre. Déléguer sa lecture à un modèle, c'est **hériter de sa situation** : de son corpus figé, de ses angles morts, de sa fraîcheur. *On échange la submersion contre une dépendance, et tout est*

de savoir ce que vaut ce à quoi l'on se rend dépendant.

Le garde-fou est inscrit dans la conception : **l'outil entretient la capacité de lire du dirigeant, il ne l'en dispense jamais.** Une IA qui résume doit laisser accéder à ce qu'elle a résumé. Une IA qui classe doit laisser revoir son classement. La probité de la délégation, c'est de connaître la situation dont on hérite — et de ne jamais prendre la lecture déléguée pour une lecture sans situation, surplombante. C'est la différence entre un solopreneur outillé et un solopreneur dépossédé.

Chapitre 9 — Entretien : maintenance, sécurité, rituels

Un flux d'observation est trop souvent conçu, déployé, puis abandonné à lui-même. Cette négligence dégrade les œuvres : un capteur dérive, un format se périmé, un canal se remplit de données que personne ne lit plus. La sixième famille d'outils — la **maintenance** — tient le canal vivant dans la durée. Elle se double d'une exigence permanente, la **sécurité**, et se structure par des **rituels** réguliers. Ce chapitre ferme la boucle des six familles.

Les outils de maintenance

Ce sont les dispositifs qui permettent de tenir un canal au fil du temps. Le canalier les **intègre dès la conception**, jamais après coup.

- **Alertes de dérive de calibration.** Un instrument de prélèvement (capteur, script, prompt d'extraction) dérive. On surveille l'écart amont/aval et l'on signale quand la mesure décroche.
- **Indicateurs de fraîcheur.** Chaque donnée porte son âge. Le système signale quand une donnée a dépassé sa fenêtre de validité au regard de la vitesse de son réel.
- **Re-prélèvement périodique.** Pour les données à péremption, on planifie le renouvellement (tâches planifiées, *cron*) : la fraîcheur ne se rattrape pas, elle se refait.
- **Contrôles de cohérence amont/aval.** On vérifie que ce qui sort du canal reste cohérent avec ce qui y entre, pour détecter une rupture silencieuse.

Pour le solopreneur, ces outils restent **sobres** : une alerte discrète, un indicateur de fraîcheur sur le tableau de bord, une tâche nocturne de consolidation. La maintenance n'est pas une usine ; c'est le soin régulier du charpentier qui passe sa charpente en revue.

La sécurité et l'hygiène numérique

La sécurité n'est pas une option, surtout pour la **donnée sensible** (coordonnées clients, données financières, données soumises au RGPD). L'**hygiène numérique** est l'ensemble des pratiques qui maintiennent les données organisées, sécurisées et conformes. Les piliers, dans la logique *local-first* :

- **Chiffrement.** Les données sensibles sont chiffrées au repos (sur le disque, dans l'archive — par exemple en AES-256) et en transit (échanges sécurisés). Une sauvegarde non chiffrée est une fuite en attente.
- **Sauvegarde 3-2-1.** Trois copies des données, sur deux supports différents, dont une hors site. C'est la règle qui protège de la **disparition** — la seconde mort de la donnée (tome 1), distincte de la péremption : ici le support se dégrade, le format devient illisible, ou l'accès est perdu. Un disque externe chiffré et une copie distante chiffrée suffisent souvent.
- **Accès restreints.** Chaque donnée n'est accessible qu'à qui en a besoin ; l'authentification protège l'accès (par exemple OAuth2 pour les services connectés).
- **Traçabilité.** Garder trace de qui a fait quoi, quand — ce que la fixation immuable (chapitre 6) facilite.
- **Minimisation et péremption réglée.** Ne prélever et ne garder que ce qui sert ; programmer la suppression au terme légal. Moins de donnée conservée, moins de surface de risque — et c'est aussi de la sobriété.

La souveraineté comme sécurité. Le meilleur garde-fou reste structurel : si la donnée brute ne quitte pas l'entreprise (traitement local, transferts maîtrisés), la surface d'attaque est réduite à la source. L'architecture sobre n'est pas seulement économe ; elle est sûre.

La réserve : le double retiré du cycle d'usage

Une donnée vit dans le cycle d'usage de l'œuvre : elle y sert, s'y consolide, y passe en péremption. Tant qu'elle y est exposée, elle peut s'y perdre — une fausse manœuvre, un support défaillant. Pour parer la perte, on lui donne un double soustrait à ce cycle : une copie **figée**, écrite une fois et jamais réécrite, qu'on ne lit pas dans l'usage courant et qu'on ne mobilise que pour rétablir ce qui a disparu. C'est la **réserve**.

La réserve n'est ni vive ni péremptée : elle est retirée du cycle d'usage — non du flux. Il n'existe aucun hors-flux (c'est la leçon de fond du tome 2) : le contenu de la réserve, comme tout prélèvement, continue de s'écarter du réel qui a poursuivi sans lui ; ce qui est suspendu, c'est son exposition à l'usage et à la fausse manœuvre, non sa péremption. On la distingue de la sauvegarde 3-2-1, qui multiplie les copies en lieux distincts pour parer le

sinistre matériel. La réserve, elle, vise la **continuité dans le temps** : elle garde, à intervalles, l'état entier de l'œuvre, et fait tourner ses **supports** — on conserve les quelques dernières générations (par exemple la courante et les deux précédentes), à plusieurs temporalités (le jour, la semaine, le mois) ; au-delà, la plus ancienne sort. Un même instantané peut servir plusieurs temporalités à la fois : on ne duplique qu'une fois par période.

Le geste du canalier est ici de **régler la cadence et le nombre de générations** selon ce qu'il peut perdre sans dommage, puis de laisser la réserve se tenir seule. Conservée par le port de stockage souverain, elle obéit aux mêmes règles que le reste : hors poste, elle patiente ; le poste revenu, elle se dépose. La réserve ne sert jamais l'activité présente — c'est sa vertu : ce qui ne sert pas ne risque pas d'être altéré par l'usage.

Les rituels de l'atelier

Un atelier vit par ses rituels — des moments réguliers, articulés, où s'opèrent les arbitrages et les soins. Sans eux, l'œuvre dérive ; avec eux, le canal reste vivant. Deux rituels structurent l'entretien.

La revue de canal

À échéance fixe (mensuelle, trimestrielle selon les cas), chaque canal entretenu est passé en revue. Ordre du jour-type :

1. Volume et qualité des prélèvements depuis la dernière revue.
2. Calibration des instruments — une dérive constatée ?
3. Pertinence des critères au regard de l'usage actuel.
4. Corpus de référence — toujours suffisant ?
5. Boucles éventuelles — le canal modifie-t-il ce qu'il observe ?
6. Signaux de péremption — quoi rafraîchir ?
7. Charge de maintenance — soutenable ?
8. Décisions et arbitrages — qui répond ?
9. **Réception effective** — le canal est-il lu par ceux à qui il est destiné, ou sa donnée reste-t-elle non lue ?

La revue de réception

Pendant de la précédente, côté lecteur. Les œuvres livrées ont-elles été **reçues** ? Leur part de lecteur était-elle praticable ? Sont-elles arrivées dans leur fenêtre de validité, ou périmées en chemin ? Quand une œuvre n'est pas lue, on diagnostique le chemin :

- **incapacité** du destinataire — chemin remédiable : on rend le client plus autonome (formation, simplification de l'interface) ;
- **désintérêt** — chemin qui appelle une décision d'attention, ou une re-conception de l'alerte ;

- **péréemption rapide** — chemin qui n'appelle aucune réponse : *toutes les données non lues ne sont pas des échecs*. Une donnée d'exercice clôturé voit sa fenêtre se fermer, et c'est sa péréemption normale.

Cette distinction épargne l'erreur la plus coûteuse : traiter comme un échec une donnée qui n'avait pas vocation à être lue, ou laisser passer comme normale une vraie non-lecture.

La signature de l'œuvre

À chaque livraison — un canal construit, une donnée consolidée, un diagnostic — le canaliier **signe**. La signature n'est pas une vanité : c'est un acte de responsabilité (l'artisan répond du travail livré) et de transparence (l'usager peut juger). Elle s'appose en tête de toute livraison, sur le modèle éprouvé par la communauté du métier.

SIGNATURE D'ŒUVRE

- Artisan : [nom]
- Date de livraison : [date]
- Type d'œuvre : [canal entretenu / donnée consolidée / diagnostic]
- Question d'usage : [la question à laquelle l'œuvre répond, en une phrase]
- Critères majeurs :
 - Périmètre : [...]
 - Cadence : [...]
 - Fidélité visée : [...]
 - Corpus mobilisé : [...]
- Optimum observationnel : [justification de l'arbitrage retenu]
- Transport / fidélité attendue à la réception :
 - [ce que le destinataire doit pouvoir faire pour que sa part soit praticable ; altérations possibles en chemin]
- Fenêtre de lecteur visée : [à qui c'est lisible, à quelles conditions]
- Limites connues : [...]
- Validité estimée : [...]
- Conditions de maintenance / re-prélèvement : [...]

Deux champs méritent l'attention, car ils ancrent la signature dans la circulation et la réception. **Transport / fidélité attendue** rappelle que signer une œuvre, c'est rendre praticable la part du lecteur — pas seulement déposer la donnée. **Fenêtre de lecteur visée** rappelle qu'une œuvre est faite pour quelqu'un de situé, et que le dire fait partie de la livraison. Sans ces deux champs, on signerait une donnée comme si elle n'avait pas à être reçue.

Documentation vivante : la maintenance par le client

Le meilleur entretien est celui que le client peut faire lui-même. La posture de référent l'exige : **documentation vivante** — chaque outil s'accompagne d'une cartographie que le client comprend, de sorte qu'il soit rédacteur en chef de ses données, non leur locataire. Une œuvre livrée sans sa documentation est une œuvre qui rend le client dépendant ; une œuvre documentée est une œuvre qu'il peut maintenir, faire évoluer, ou confier à un autre. C'est le **critère vérifiable** du métier, appliqué

jusqu'à la dernière étape : à tout moment, le client reprend la main sans rien perdre.

Chapitre 10 — L’atelier numérique : le matériel et les logiciels

Un artisan sans atelier n’est pas un artisan : c’est un théoricien de l’artisanat. L’atelier est le lieu où le geste se fait, où l’outil est tenu, où l’œuvre se montre. Ce chapitre décrit l’établi du canalier numérique : le **matériel** (hardware) et les **logiciels** (software) de 2026. Tout y est daté, donc révisable ; ce qui ne l’est pas, c’est le principe qui guide chaque achat — *au juste besoin, ni aveugle ni sur-ingénierié*.

Le matériel

Le métier de canalier est à **faible intensité capitalistique** : il ne demande ni machine rare ni gros investissement. Quelques milliers d’euros suffisent à équiper un atelier complet. C’est une force du métier — on s’installe sans s’endetter.

Le poste de travail

- **Un ordinateur portable** correct et mobile : c’est l’outil central, qui sert à développer, à se déplacer chez le client, à faire tourner les modèles d’IA légers. Un processeur récent et une mémoire vive généreuse comptent plus qu’une puissance graphique extrême. La mobilité permet la **proximité** — aller chez le client, prélever sur place, former en présence.
- **Un ou deux écrans** : pour tenir le geste juste, lire le code et l’œuvre côte à côte.
- **Un poste ergonomique** (bureau, fauteuil) : le geste de lecture d’écran est long ; le corps de l’artisan est aussi un outil.

Le serveur

Selon les besoins du client et de l’atelier, un **serveur** local prend plusieurs visages, parfois cumulés sur une même petite machine :

- **Serveur web** — pour héberger les applications de pilotage, en local chez le client ou sur une petite machine d'atelier.
- **Serveur de base de données** — quand le volume ou le partage dépasse ce qu'un simple fichier local peut tenir.
- **Serveur LLM local** — une machine qui sert les modèles d'IA en interne, pour que la donnée ne quitte pas l'entreprise. Un serveur modeste suffit pour des modèles légers ; une carte graphique correcte accélère les modèles plus capables.

L'architecture **edge / local-first** se décline sur un continuum : du simple terminal du client (où tout tient sur son appareil) à un petit serveur d'atelier mutualisé entre plusieurs clients d'un même territoire. On dimensionne au plus près du besoin réel.

Le stockage et la sauvegarde

- **Un serveur NAS** ou un stockage en réseau : pour centraliser et sauvegarder les œuvres de l'atelier.
- **Un ou plusieurs disques externes chiffrés** : pour la règle **3-2-1** (trois copies, deux supports, une hors site). C'est la garantie contre la disparition de la donnée.

Les périphériques

Une imprimante (souvent professionnelle, pour les livrables et la documentation), un scanner si l'on numérise des documents existants. Modestes, mais utiles à la circulation et à l'archivage.

Budget-type d'installation. L'expérience des entreprises pilotes situe l'équipement initial complet (portable, écran, disque externe, petit serveur/NAS, périphériques, poste ergonomique) dans une fourchette de quelques milliers d'euros — un profil financier robuste et peu risqué, souvent allégé par des aides à la création.

Les logiciels

Le software du canalier est, autant que possible, **ouvert, gratuit et pérenne** — cohérent avec la souveraineté qu'il vend à ses clients.

L'environnement de développement

- **Un système d'exploitation libre** : une distribution Linux (par exemple Debian) offre stabilité, gratuité et maîtrise. Elle équipe aussi bien le poste que les serveurs.
- **Un éditeur de code (IDE)** : l'outil où s'écrit l'œuvre. On en choisit un qui supporte le travail multi-fichiers de l'architecture hexagonale, le contrôle de version, et l'assistance par IA si on le souhaite.

- **Un gestionnaire de versions (Git)** et un hébergement de dépôt : pour garder l'histoire du code, revenir en arrière, et conserver une trace des évolutions. Git est à l'atelier ce que le registre immuable est à la donnée : la mémoire fiable du travail.

La conteneurisation

- **Docker et Docker Compose** : pour emballer une application et ses services (base, cache, file) dans des **conteneurs** reproductibles. Le bénéfice est décisif pour le canalier : ce qui tourne sur son poste tourne à l'identique chez le client, et les **environnements** se séparent proprement (Développement, Test, *Staging*, Production). On livre un assemblage qui s'installe et se réinstalle sans surprise.

Le langage et les services

- **JavaScript / Node.js** comme langage principal : un seul langage du cœur métier jusqu'aux scripts d'extraction et aux services, ce qui simplifie la maintenance pour un atelier d'une personne.
- **Bases de données ouvertes** : *SQLite* (un fichier, sans serveur, idéal pour la sobriété) ou *PostgreSQL* (serveur, pour le volume et le partage).
- **Services internes** au besoin, et seulement au besoin : un cache rapide (Redis), une file de messages (RabbitMQ), un service d'authentification (OAuth2), une agrégation de logs. Ces briques se branchent en **adaptateurs** ; on ne les ajoute qu'au franchissement réel d'un seuil de complexité.

L'IA locale

- **Un runtime de modèles locaux** (de type *Ollama*) et quelques **modèles ouverts légers** (familles Llama, Phi, Gemma...), choisis au juste besoin de chaque tâche (classer, extraire, rédiger, résumer). Voir le chapitre 8 pour leur emploi.

La documentation et la production de livrables

- **Markdown** pour écrire la documentation vivante et les livrables textuels ; **pandoc** pour en produire des PDF propres. Du texte brut, durable, versionnable — qui ne dépend d'aucun traitement de texte propriétaire.

L'atelier comme lieu, physique ou virtuel

Au-delà des machines, l'atelier est un **lieu** doté de fonctions que les outils seuls ne donnent pas :

- **un seuil** — un espace identifié où l'on entre pour *faire* la donnée (à l'extérieur, on la consomme) ;
- **un espace de travail** où poser ses instruments et ses œuvres en cours ;
- **une bibliothèque** — la mémoire de l'atelier : ouvrages de référence, grilles d'annotation conservées, archives des flux passés, documentation des œuvres livrées ;
- **un mur de l'œuvre exposée** — pour honorer le travail, enseigner par l'exemple, donner des points de référence.

Quand l'atelier est **virtuel** (espace de travail en ligne, dépôt partagé), il doit reproduire ces **fonctions**, pas seulement ranger des fichiers : un seuil d'accès identifié, des zones personnelles, un espace de revue, des archives organisées, une galerie des œuvres. Sans cela, ce n'est plus un atelier, c'est un dossier partagé.

L'articulation atelier / chaîne

Dans toute organisation, l'atelier coexiste avec la **chaîne** — le traitement massif, à bas coût, des flux standardisés. Les deux sont nécessaires et ne servent pas les mêmes besoins. La chaîne traite les volumes courants ; l'atelier fait **à la main** les flux qui le méritent : ceux qui guident des décisions à fort enjeu, ceux qui doivent durer, ceux dont la traçabilité compte, ceux qu'il faut rendre lisibles à un destinataire précis. Le passage de l'un à l'autre se décide explicitement. Le canalier sait quand sortir l'établi, et quand laisser faire la chaîne. C'est encore un arbitrage à l'optimum — le dernier, et le plus stratégique.

Chapitre 11 — Parcours d'équipement : assembler les outils

Les chapitres précédents ont décrit les outils un à un. Celui-ci les assemble. Car un outil ne vaut que dans un geste, et un geste que dans un parcours. On montre ici comment les méthodes (hexagonale, gamification, cœur natif, IA locale) et les six familles d'outils s'enchaînent au service d'un solopreneur — du premier contact à l'œuvre entretenue. Tout y est illustration : un autre client appellera un autre assemblage.

Le procédé, toujours le même : trois gestes

Quel que soit le client, le canalier suit trois gestes, dans l'ordre.

1. **Modéliser avant de coder.** Parler le métier du client, distinguer son **cœur de métier** (intouchable) de sa **matrice de soutien** (automatisable), et modéliser la matrice pour libérer le cœur.
2. **Prélever au juste besoin.** Situer l'optimum entre sous-prélèvement et sur-ingénierie, à reposer à chaque mission.
3. **Livrer une œuvre lisible et réversible.** Documentée, comprise du client, signée, et dont il peut à tout moment reprendre la main.

Le développement procède par étapes, jamais d'un bloc : **imprégnation** (installer le socle vital), **ajustement** (le client apprivoise et identifie ses vrais besoins), **expansion** (on ajoute des modules si le besoin le justifie).

Le parcours type, étape par étape

Étape 0 — L'observation

Une phase d'**audit** gratuite ou légère : échanger avec le dirigeant sur son fonctionnement, ses attentes, ses problèmes, ses opportunités. Cartographier son processus métier (étapes, tâches, séquences). Repérer les tâches chronophages, pénibles, sources d'erreurs. Livrable : un état

des lieux et un plan d'action priorisé. *On ne propose rien avant d'avoir écouté.*

Étape 1 — La schématisation (méthode hexagonale)

Schématiser le système de gestion des données selon l'**architecture hexagonale** : isoler le cœur métier, nommer les ports, identifier les adaptateurs (stockage, interface, services tiers, capteurs). Le client obtient une vision claire de ce qu'on va construire, et peut en évaluer délais, budget, gains. C'est ici que se décide la pérennité.

Étape 2 — La logique métier (en clair, puis en code)

Rédiger les **entités, règles et cas d'usage** en langage clair (chapitre 2), les faire valider par le client, puis les implémenter — en **JavaScript natif** (chapitre 3). Le cœur est testable hors de toute technologie.

Étape 3 — Le prélèvement et la fixation

Mettre en place les **outils de prélèvement** au juste besoin (formulaires mobiles, scripts d'extraction, capteurs, extraction sémantique par IA locale) et les **grilles d'annotation** (chapitre 5). Choisir le **support de fixation** ouvert et durable (chapitre 6) : souvent un stockage **local-first** sur l'appareil du client, avec chiffrement.

Étape 4 — Le pilotage (lecture + gamification)

Construire le **tableau de bord** (chapitre 7) : trois questions — *où en suis-je, que dois-je faire, qu'est-ce qui dérive* —, des indicateurs choisis avec le client, des **seuils** (fiscaux, métier, objectifs). L'habiller de **gamification** honnête (chapitre 4) pour qu'il soit lu et agréable : barres de progression, alertes justes, anticipation des seuils. Le client passe d'une activité subie à une trajectoire pilotée.

Étape 5 — La circulation distante (si besoin)

Si le client doit partager (comptable, partenaire, agent IA), ajouter les **adaptateurs de transport** : exports CSV/PDF consolidés, ou une **API (au standard MCP)** pour les agents d'IA — sans toucher au cœur. La part du lecteur destinataire doit être praticable.

Étape 6 — L'entretien et l'autonomie

Mettre en place la **maintenance** (alertes de fraîcheur, sauvegarde 3-2-1, re-prélèvements), instaurer les **rituels** (revue de canal, revue de réception), **signer** l'œuvre, et — surtout — livrer la **documentation vivante** qui rend le client autonome. La mission récurrente n'est pas une

dépendance entretenue : c'est le forgeron qu'on rappelle pour une pièce nouvelle.

Trois assemblages illustrés

A. L'artisan du bâtiment qui doit basculer vers la facturation certifiée

Besoin : un peintre, un maçon habitués à un tableur doivent passer à une solution de facturation conforme, sans savoir choisir. *Assemblage* : peu de développement, beaucoup d'accompagnement. On audite, on présente les options, on installe et paramètre, on **forme** — idéalement en **format mutualisé** (un petit groupe partage le besoin, échange, progresse ensemble). L'œuvre n'est pas un logiciel : c'est une autonomie. Outils mobilisés : audit, grille de choix, formation.

B. Le solopreneur mobile qui pilote son stock et sa trésorerie

Besoin : remplacer un cahier oublié par un pilotage à jour. *Assemblage* : cœur hexagonal en JS natif ; **PWA installable** sur le téléphone, fonctionnant hors ligne ; stockage **local-first** chiffré ; saisie fondue dans le geste (chaque mouvement met à jour un indicateur) ; tableau de bord gamifié avec anticipation du seuil de TVA ; sauvegarde 3-2-1 ; export PDF pour le comptable. L'artisan ne « remplit plus un tableau » : il dispose d'un inventaire et d'une trésorerie lisibles d'un coup d'œil.

C. L'entreprise qui veut des indicateurs sur mesure dans un gros logiciel

Besoin : un dirigeant possède un logiciel coûteux mais n'en tire pas ses indicateurs. *Assemblage* : aucun remplacement. Le canalier écrit des **requêtes** sur la base existante (prélèvement par extraction), que le logiciel affiche. Éventuellement, une logique métier d'aide à la décision (par exemple la performance unitaire de chaque véhicule d'une flotte), rédigée en clair puis implémentée. Outils mobilisés : extraction, logique métier hexagonale, lecture. Prélever, parfois, c'est savoir interroger ce qui est déjà là.

Le fil qui tient tout

À travers ces assemblages, un même fil : **donner la capacité, puis le choix**. Le canalier ne vend pas un produit qui enferme ; il transmet une capacité qui libère. Chaque outil de ce manuel — du plus humble formulaire au modèle d'IA local — est jugé à la même aune : *rend-il le client plus maître de son activité, ou l'en éloigne-t-il ?* Tant que la réponse est la première, l'œuvre est juste.

L'artisan seul ne pèse rien sur l'économie de la donnée ; le maillage, oui. Chaque solopreneur outillé, rendu capable de lire et de partager sa donnée sur son territoire, est une maille. La puissance vient de leur fédération — mais elle commence, toujours, par un canal bien creusé, à l'optimum, signé, lisible, et qu'un client peut reprendre en main. C'est l'objet de ce manuel ; le reste est affaire de pratique, et de la main qui apprend.

La fédération n'exige pas de centre — elle n'exige qu'une maille de plus, dont le métier est de coordonner. Le maître d'œuvre d'un chantier ne commande pas les plannings des corps de métier : il tient la lecture la plus complète, tire les dépendances, et propage par l'échange ce que chacun accepte dans sa propre œuvre. Coordonner par l'échange coûte moins de gestes que le maillage nu — et produit ce que le maillage nu ne produit pas : l'ordre cohérent. C'est un métier, pas une autorité ; le jour où il devient une autorité, le maillage est redevenu une pyramide.

Chapitre 12 — L’atelier d’agents IA : industrialiser le parcours

Le chapitre précédent a décrit le parcours d’équipement tel qu’un canalier le conduit à la main : observer, schématiser, modéliser, prélever, piloter, faire circuler, entretenir. Ce chapitre décrit comment **automatiser** ce même parcours, sans rien changer à ses gestes, en le confiant à une équipe de modèles d’IA spécialisés que l’on nommera les **Agents IA**. Le canalier sait les gérer : c’est son établi, augmenté.

Comme tout ce manuel, ce chapitre est de la **forme**. Les outils qu’il nomme — un agent de codage, un moteur de conteneurs, un hébergeur de dépôt — sont datés. Le fond qu’il sert — le parcours, les trois gestes, la posture de référent — ne se périmé pas.

Le socle technique : ne pas refaire l’éprouvé

Une part de l’œuvre ne change jamais d’une activité à l’autre : la tuyauterie — servir des fichiers, fonctionner hors ligne, se mettre à jour, ranger l’état. Le canalier ne la réécrit pas à chaque fois ; il la tient pour **socle éprouvé** et n’engendre que le domaine. C’est la part déterministe du parcours : moins l’atelier génère, moins il se trompe.

Sur ce socle, une règle simple : **une dépendance externe est une dette et un risque**. Quand une primitive intégrée et éprouvée fait l’affaire, on la préfère — le chiffrement par l’interface native du navigateur plutôt qu’une bibliothèque tierce chargée d’ailleurs, un serveur de quelques lignes plutôt qu’un paquet à installer. Sur ce qui est sensible, cette préférence n’est pas négociable : une dépendance peut être compromise, et l’œuvre doit pouvoir tourner sans rien installer, sans réseau, et rester réversible. Un seul système de modules, des liens cohérents, des essais qui passent avant toute remise : voilà ce qui distingue une œuvre dont le client reprend la main d’un assemblage qui se défait.

Le détail applicable — quelles interfaces, quel format, quelles vérifications — relève de la forme et vit hors de ce manuel, dans les contraintes techniques que lisent les agents. Le principe, lui, ne se périmé pas.

Les Agents IA

Un Agent IA est un modèle d'IA tenu par une **charte** : un rôle, un artefact qu'il lit, un artefact qu'il produit, des contraintes, et ce qui doit être vrai avant qu'il passe la main. À chaque étape du parcours correspond un agent. Ils travaillent en chaîne : non pas un document qui s'allonge, mais une **série de documents distincts**, versionnés et lisibles par un humain, chacun l'ordre de travail du suivant.

La singularité du procédé tient en une phrase : **les documents amont ne sont pas de simples spécifications techniques — ce sont les chapitres du manuel qualité de l'activité**. Le manuel d'activité n'est pas produit à côté du logiciel ; il est la part amont de la chaîne. D'où une frontière, ici opérante :

- au-dessus de la ligne, la description de l'activité (acteurs, produits, processus, traçabilité, flux, données) : c'est le manuel, le **fond**, l'actif durable ;
- en dessous, le dossier hexagonal, le code, les tests : **forme datée**, régénérable à partir du fond.

L'actif n'est pas le logiciel. C'est la spécification de l'activité.

Deux objets à ne pas confondre : l'atelier d'agents IA et l'œuvre

L'atelier d'agents IA est l'atelier de production. Il réunit, sur la machine locale, le moteur de modèles, les agents, un bac à sable d'exécution et de test, un dépôt. On peut l'isoler dans un conteneur (Docker) : c'est l'outil de production, opéré par l'artisan.

L'œuvre est le logiciel produit. Elle obéit aux chapitres 2 et 3 : un cœur natif hexagonal, local-first, installable, réversible, que le client peut reprendre. **Elle n'exige pas l'atelier d'agents IA pour fonctionner**. On ne livre pas un atelier à un client : on lui livre un meuble dont il a les plans. Confondre les deux — demander à l'entrepreneur d'opérer au quotidien la machinerie qui a servi à fabriquer — serait trahir le geste de réversibilité.

L'agent de codage : l'établi des agents

Les Agents IA ont besoin d'un environnement qui leur donne, sous supervision, l'accès au système de fichiers, au shell et aux outils : c'est le rôle d'un **agent de codage**. En 2026, on emploie pour cela un outil tel que Vibe Code : il lit les fichiers, exécute des commandes, écrit du code, le tout sous le contrôle de l'opérateur. Rien ne s'écrit sans supervision — ce qui fait du jalon de validation humaine non pas une option, mais la mécanique même de l'outil. Cet outil est de la forme : il sera remplacé.

Le principe qu'il sert — des agents qui agissent sur l'établi sous l'œil du référent — demeure.

Le dépôt : sauvegarde, mémoire, et base distribuée

Le travail est versionné dans un dépôt **git**, pour deux raisons déjà connues du chapitre 9 : la **sauvegarde** (l'œuvre ne tient pas à un seul disque) et la **mémoire** (l'évolution se relit, se compare, se défait). À cela s'ajoute une troisième fonction, propre à l'atelier d'agents IA : le dépôt sert de **base distribuée**. L'ensemble nécessaire au lancement — le corpus comme base de connaissance, les chartes, les gabarits, la procédure de mise en route — tient dans un dépôt que l'on clone ou télécharge, et que l'on copie sur la machine locale. L'artisan dispose ainsi d'une base complète, prête à l'emploi, qu'il fait évoluer pour son territoire.

Les ports souverains : étendre sans dépendre

Le modèle local suffit à la plupart des tâches, et il est souverain par construction. Là où il ne suffit pas, l'atelier — comme l'œuvre — fait travailler des **services souverains performants**, branchés par l'hexagone (chapitres 3 et 6, « connecter le monde extérieur — toujours par l'hexagone »). Trois usages se présentent :

- un **modèle frontalier souverain** pour les tâches sémantiques précises hors de portée du local ;
- la **prévision** par modèles de fondation pour séries temporelles (demande, stocks, charge) ;
- les **sources de données inter-professionnelles**, pour la veille collective.

Règle absolue, héritée de la souveraineté distribuée : ces ports sont **non vitaux**. Le cœur fonctionne quand le lien extérieur se rompt ; le service souverain est un enrichissement, jamais une dépendance. Un adaptateur, donc, et toujours substituable.

Le pilotage de l'atelier : déterministe d'abord, l'humain aux jalons

Un modèle local est plus faible qu'un modèle frontalier pour produire du code fiable. Le chapitre 8 a donné la parade : **combiner l'IA et les règles déterministes**. L'atelier la met en œuvre à l'échelle du logiciel entier. Il est **majoritairement déterministe** — gabarits, échafaudage, génération bornée des seuls écarts sur un socle pré-construit et testé — et ne mobilise le modèle que là où il excelle : comprendre le métier, cartographier, rédiger. Ce n'est pas une autonomie totale ; c'est un parcours assisté à points de contrôle, où le référent valide avant chaque passage de main.

La mise en pratique

Le détail concret — structure du dossier, documents à fournir, ordre de lancement des agents, commandes — relève d'un **get-started** vivant, tenu dans le dépôt et révisé avec lui. Ce manuel ne le reprend pas : un mode d'emploi qui nomme des commandes et une adresse se périmerait en quelques saisons, et un livre de fond n'a pas à vieillir au rythme d'un outil. Le manuel dit le principe ; le dépôt dit le geste du jour.

Ce qui, dans ce chapitre, ne se périme pas

L'agent de codage, le moteur de conteneurs, l'hébergeur de dépôt, les services souverains nommés : tout cela passera. Ce qui demeure, c'est qu'un parcours d'équipement peut être confié à des agents tenus par des chartes ; que le manuel qualité de l'activité en est la part durable et le logiciel la part régénérable ; que l'on ne livre pas l'atelier mais l'œuvre ; et qu'aucun service extérieur ne doit jamais devenir vital. Le canalier qui tient cela tiendra n'importe quel outil que demain lui présentera.

Annexe — La boîte à outils du canalier

Cette annexe rassemble, en référence rapide, les outils de questionnement et les repères du manuel. Un outil n’est pas une règle : la règle dispense de penser, l’outil ouvre une question. Aucune de ces fiches ne rend de verdict automatique — elles servent à *ne rien oublier d’essentiel*, jamais à remplacer le jugement situé.

A. Les six familles d’outils, en un coup d’œil

Temps de la donnée	Famille	Outils numériques 2026
Fabrication	Prélèvement	Formulaires mobiles, scripts d’extraction, capteurs/IoT, IA locale (extraction sémantique)
Fabrication	Annotation	Grilles et classifications, protocoles, annotation assistée par IA
Circulation	Fixation	Formats ouverts (JSON, CSV, SQLite), immuabilité (UUID v7, hash), archivage chiffré
Circulation	Transport	Local : consolidation, base, cache, files. Distant : export CSV/PDF, API (MCP)
Réception	Lecture	Tableau de bord natif (HTML/CSS/JS), indicateurs, visualisations sobres, gamification
Entretien	Maintenance	Alertes de fraîcheur/calibration, re-prélèvement, sauvegarde 3-2-1, contrôles de cohérence

B. Les méthodes transversales

- **Architecture hexagonale** — isoler le cœur métier ; ports & adapters ; changer la forme sans toucher au fond ; modéliser avant de coder.

- **Cœur sans dépendance** — HTML/CSS/JS natif ; SPA/PWA ; local-first ; pérennité, légèreté, réversibilité.
- **Gamification UX** — rendre la donnée consolidée lisible et engageante ; seuils ; barres, badges, alertes justes ; garde-fou : récompenser le geste juste, jamais le chiffre brut.
- **IA locale** — opérateur sémantique ; souveraineté, coût, maîtrise ; l’IA propose, l’humain décide ; combiner avec les règles déterministes.

C. Checklist du flux d’observation rencontré

À se poser devant un jeu de données, un indicateur, un chiffre.

Fabrication — 1. De quel flux du réel cette donnée provient-elle ? 2. Qui l’a creusée, dans quelle intention ? 3. Quels critères — cadence, périmètre, instrument ? 4. Quelles unités, normes, références implicites ? 5. Qu’est-ce qui n’a pas été prélevé et pourrait être pertinent ? 6. Quelle est la date du prélèvement, et son âge fonctionnel ? 7. Le flux est-il dans une boucle — modifie-t-il ce qu’il observe ? 8. Quels biais la classe d’instrument engendre-t-elle ?

Circulation — 9. La chaîne amont est-elle traçable jusqu’à la source ? Par combien de mains, de copies, de retraitements — et qu’a-t-elle pu y subir ?

Réception — 10. Pour mon usage, ce flux est-il adapté ? 11. Ai-je le moyen et le temps de le lire, au regard de ma fenêtre ? Sinon, est-ce incapacité, désintérêt ou péremption rapide ?

D. Grille de péremption

Pour chaque donnée, situer sa temporalité d’usage (déclencheur de question, non verdict). Une donnée périssable n’est pas une mauvaise donnée : c’est une donnée à fenêtre de validité courte.

Donnée	Fragment du réel	Vitesse de changement	Âge fonctionnel	Chemin probable de non-lecture
Donnée démographique	Population	Lente	Plusieurs années	—
Indicateur économique	Marché	Rapide	Jours à semaines	Péremption rapide
Mesure environnementale	Air, eau	Variable	Heures à mois	Selon usage
Score comportemental	Personne	Rapide	Jours	Péremption rapide

E. Modèle de revue de canal (ordre du jour)

1. Volume et qualité des prélèvements.

2. Calibration des instruments — dérive ?
3. Pertinence des critères au regard de l'usage.
4. Corpus de référence — suffisant ?
5. Boucles — effets observés ?
6. Signaux de péremption — quoi rafraîchir ?
7. Charge de maintenance — soutenable ?
8. Décisions et arbitrages — qui répond ?
9. Réception effective — lu, ou non lu (incapacité / désintérêt / péremption rapide) ?

F. Fiche de signature d'œuvre

À apposer en tête de toute livraison (détail au chapitre 9).

SIGNATURE D'ŒUVRE

- Artisan : [nom]
- Date de livraison : [date]
- Type d'œuvre : [canal entretenu / donnée consolidée / diagnostic]
- Question d'usage : [la question à laquelle l'œuvre répond, en une phrase]
- Critères majeurs :
 - Périmètre : [...]
 - Cadence : [...]
 - Fidélité visée : [...]
 - Corpus mobilisé : [...]
- Optimum observationnel : [justification de l'arbitrage retenu]
- Transport / fidélité attendue à la réception :
 - [ce que le destinataire doit pouvoir faire pour que sa part soit praticable ; altérations possibles en chemin]
- Fenêtre de lecteur visée : [à qui c'est lisible, à quelles conditions]
- Limites connues : [...]
- Validité estimée : [...]
- Conditions de maintenance / re-prélèvement : [...]

G. Aide-mémoire matériel et logiciel

Matériel — ordinateur portable, écran(s), poste ergonomique ; serveur (web / base / LLM local) ; NAS et disques externes chiffrés (sauvegarde 3-2-1) ; imprimante/scanner.

Logiciel — OS libre (Linux Debian) ; IDE + Git ; Docker & Docker Compose (environnements Dev/Test/Staging/Prod) ; Node.js/JavaScript ; SQLite ou PostgreSQL ; au besoin Redis, RabbitMQ, OAuth2 ; Ollama + modèles ouverts légers (Llama/Phi/Gemma) ; Markdown + pandoc.

H. Les garde-fous, en sept phrases

1. L'optimum se repose à chaque mission : ni aveugle, ni noyé.
2. Le cœur métier ne dépend d'aucune technologie ; la forme est remplaçable, le fond ne l'est pas.
3. Traiter au plus près de la source ; ne faire voyager que ce qui doit voyager.

- 4. L'IA propose, l'humain décide ; la validation reste humaine sur tout ce qui est critique.
- 5. Gamifier le geste juste, jamais le chiffre brut ; ne pas maquiller une projection en mesure.
- 6. L'outil entretient la capacité de lire du client, il ne l'en dispense jamais.
- 7. À tout moment, le client peut reprendre la main sans rien perdre.

I. Table de correspondance — le vocabulaire du corpus et les cadres existants (forme datée, juillet 2026)

Le fond ne doit rien aux standards du moment ; le praticien, lui, les rencontrera. Cette table, révisable et datée, donne les équivalences de travail — sans identité de sens.

Corpus	Cadre existant	Ce qui coïncide / ce qui diffère
Chaîne de traçabilité ; acte d'inscription attribué	Provenance / lineage (W3C PROV)	Même souci de remonter la chaîne ; PROV modélise des entités et activités, le corpus part de l'acte et de son sujet.
Fixation immuable ; passage d'état écrit avant retrait	Event sourcing ; journaux append-only	Même primat de l'événement inscrit sur l'état courant ; le corpus y ajoute la péremption d'état réglée par l'usage.
Part du lecteur praticable ; corpus de référence joint	Principes FAIR (findable, accessible, interoperable, reusable)	FAIR outille la circulation ; le corpus rappelle que la réception ne se décrète pas — une donnée FAIR peut rester muette.
Fraîcheur ; durée limite d'utilisation	TTL, data freshness / staleness	Même mécanique de seuil ; le corpus fonde le seuil dans l'usage du lecteur, non dans la technique.
Convenu ; sources stables entretenues	Référentiels, vocabulaires contrôlés, master data	Même fonction d'accord partagé ; le corpus insiste : entretenu, donc périssable.

Toute théorie gagne en pratique

Pour le canalier qui veut garnir son établi, pour l'apprenti qui apprend les gestes, pour le solopreneur qui veut comprendre ce que son outil fait – et pourquoi.

On ne se baigne jamais deux fois dans le même fleuve. Mais on peut apprendre à en creuser des canaux. Après la trilogie *À l'épreuve du flux*, qui disait le fond, *Données techniques* donne la forme : le **manuel technique du canalier**.

Outils et méthodes pour creuser, aujourd'hui, un canal numérique — bâtir du durable, fabriquer la donnée, la faire circuler, la rendre lisible et l'entretenir. Au service d'un seul but : offrir au solopreneur une gestion fluide, lisible, maîtrisée, et la sérénité qui vient avec. Non un logiciel dont on ne peut plus sortir, mais une capacité qui libère.

Edition juin 2026

Les outils changent ; le métier demeure